CHAT

V2 IMAGE-2 BASED EDITION

CHATGPT HYPERKB OS

AGI-Oriented Knowledge Operating System

OPERATING THESIS

This volume turns ChatGPT from a passive chatbot into a governed cognitive workstation: prompt kernel, project memory, research loop, canvas revision, data analysis, web freshness, coding support, multimodal reasoning, safety courts, business adoption, agentic execution, and versioned learning. It is not an official OpenAI manual; it is an operational HyperKB synthesis anchored to official OpenAI product and help sources and designed for conversion, teaching, team deployment, and high-trust execution.

CONVERSION PROMISE

Reader promise: learn exactly how to use ChatGPT as a serious system for research, writing, business, learning, coding, data, privacy-aware workflows, and agentic execution without confusing feature hype with verified capability.

IMAGE-2 DESIGN DNA

Visual language: cream grid, dark left spine, capsule headers, pop-flat abstract signals, colored courts, boxed mechanisms, no empty page luxury, and page chunks optimized for retrieval and rereading.

FINAL USE

Use this PDF as a prompt library, course book, internal AI adoption guide, RAG chunk source, GraphRAG node map, consulting framework, and model-evaluation checklist. Verify live product claims against official OpenAI release notes before commercial publication.

Zouhair Moussaif / Moussaif HyperKB Series

450 pages | generated 2026-07-03

Official OpenAI Source Rank

Official | ChatGPT | Product facts | Verify live

SAFETY DECLARATION

Available material: official OpenAI web pages discovered during generation, prior Moussaif HyperKB style constraints, and the visible image-2 layout DNA. Directly sourced: product existence, release-note practice, projects, canvas, data controls, privacy, enterprise privacy, search, pricing, and model release notes. Inferred: the OS architecture, prompt courts, HyperLoops, and training protocols. Speculative: AGI-oriented framing and MCD/QMCD evaluation. Needs verification: live UI, plan limits, country availability, rollout state, and workspace permissions.

NON-OFFICIAL STATUS

This PDF is not an OpenAI publication and must not be marketed as official documentation. It is a synthesis and operating manual for serious use of ChatGPT. Every feature-specific operational claim should be checked against official OpenAI release notes and help pages before deployment.

SOURCE LEDGER

1. OpenAI ChatGPT product page - Official product positioning for ChatGPT as an AI chatbot for everyday use. URL: https://openai.com/chatgpt/
2. OpenAI About - Official mission statement and AGI orientation. URL: https://openai.com/about/
3. OpenAI Help Center - ChatGPT collection - Official navigation hub for ChatGPT capabilities, controls, and support articles. URL: https://help.openai.com/en/collections/3742473-chatgpt
4. ChatGPT Release Notes - Rolling official record of changes, launches, and availability notes. URL: https://help.openai.com/en/articles/6825453-chatgpt-release-notes
5. Projects in ChatGPT - Official explanation of projects, context, tools, and project memory behavior. URL: https://help.openai.com/en/articles/10169521-projects-in-chatgpt
6. Canvas in ChatGPT - Official explanation of the writing/coding revision workspace and inline feedback. URL: https://help.openai.com/en/articles/9930697-what-is-the-canvas-feature-in-chatgpt-and-how-do-i-use-it
7. Data Controls FAQ - Official description of controls over whether conversations help improve models. URL: https://help.openai.com/en/articles/7730893-data-controls-faq

SOURCE LEDGER CONTINUED + CLAIM GOVERNANCE

8. Consumer Privacy - Official privacy controls including temporary chats, memory controls, and security framing. URL: https://openai.com/consumer-privacy/
9. Enterprise Privacy - Official enterprise commitments around ownership and control of business data. URL: https://openai.com/enterprise-privacy/
10. Introducing ChatGPT search - Official description of web search integration and timely answers with sources. URL: https://openai.com/index/introducing-chatgpt-search/
11. ChatGPT Pricing - Official plan and capability differences; verify before quoting limits or availability. URL: https://openai.com/pricing
12. ChatGPT usage and adoption patterns at work - Official business report on workplace usage patterns and synthetic examples. URL: https://openai.com/business/guides-and-resources/chatgpt-usage-and-adoption-patterns-at-work/
13. Model Release Notes - Official model-side changes; use for version-sensitive claims. URL: https://help.openai.com/en/articles/9624314-model-release-notes

Claim rule: Fact = official page. Interpretation = synthesis. Tool = operational construction. Metaphor = design language. Hypothesis = testable workflow claim. Proposal = recommended practice, not product guarantee.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Interface Identity - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats interface identity inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about interface identity, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what interface identity must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for interface identity: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for interface identity. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Assistant Boundary - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats assistant boundary inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about assistant boundary, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what assistant boundary must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for assistant boundary: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

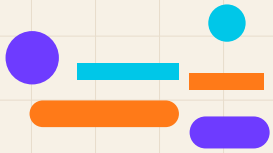
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for assistant boundary. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Model Versus Product - Agentic Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats model versus product inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about model versus product, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what model versus product must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for model versus product: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for model versus product. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Capability Map - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats capability map inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about capability map, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what capability map must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for capability map: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for capability map. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Plan Variance - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats plan variance inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about plan variance, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what plan variance must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for plan variance: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for plan variance. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Release-Note Dependency - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats release-note dependency inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about release-note dependency, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what release-note dependency must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for release-note dependency: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for release-note dependency. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

User Intent - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats user intent inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about user intent, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what user intent must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for user intent: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for user intent. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Trust Contract - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats trust contract inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about trust contract, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what trust contract must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for trust contract: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for trust contract. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Conversation State - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats conversation state inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about conversation state, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what conversation state must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for conversation state: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

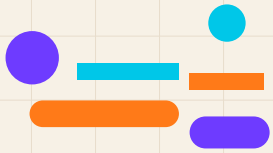
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for conversation state. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Answer Versus Action - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats answer versus action inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about answer versus action, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what answer versus action must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for answer versus action: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for answer versus action. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Tool Surface - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats tool surface inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about tool surface, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what tool surface must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for tool surface: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for tool surface. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Source Governance - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats source governance inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about source governance, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what source governance must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for source governance: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for source governance. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Prompt-As-Program - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats prompt-as-program inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about prompt-as-program, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what prompt-as-program must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for prompt-as-program: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

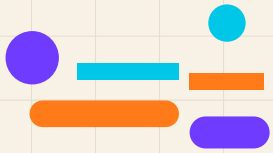
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for prompt-as-program. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Knowledge Limit - Evaluation Engine



ChatGPT | MCD | OS chunk | Court-tested

OPERATING THESIS

This chunk treats knowledge limit inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about knowledge limit, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what knowledge limit must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for knowledge limit: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for knowledge limit. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Task Framing - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats task framing inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about task framing, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what task framing must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for task framing: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for task framing. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Context Debt - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats context debt inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about context debt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what context debt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for context debt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for context debt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Chat As Workspace - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats chat as workspace inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about chat as workspace, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what chat as workspace must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for chat as workspace: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for chat as workspace. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Identity Of Response - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats identity of response inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about identity of response, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what identity of response must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for identity of response: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for identity of response. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Confidence Calibration - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats confidence calibration inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about confidence calibration, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what confidence calibration must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for confidence calibration: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for confidence calibration. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Workflow Role - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats workflow role inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about workflow role, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what workflow role must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for workflow role: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for workflow role. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Ai Literacy - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats AI literacy inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about AI literacy, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what AI literacy must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for AI literacy: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for AI literacy. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Human Agency - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats human agency inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about human agency, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what human agency must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for human agency: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for human agency. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Product Drift - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats product drift inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about product drift, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what product drift must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for product drift: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for product drift. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Session Boundaries - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats session boundaries inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about session boundaries, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what session boundaries must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for session boundaries: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for session boundaries. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Feature Discovery - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats feature discovery inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about feature discovery, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what feature discovery must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for feature discovery: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for feature discovery. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Misuse Boundaries - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats misuse boundaries inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about misuse boundaries, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what misuse boundaries must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for misuse boundaries: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for misuse boundaries. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Operational Thesis - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats operational thesis inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about operational thesis, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what operational thesis must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for operational thesis: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for operational thesis. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Final Constitution - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats final constitution inside ChatGPT Ontology & Product Identity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about final constitution, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what final constitution must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for final constitution: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with ChatGPT Ontology & Product Identity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a ChatGPT Ontology & Product Identity operating assistant for final constitution. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Prompt Kernel - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats prompt kernel inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about prompt kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what prompt kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for prompt kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for prompt kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Instruction Hierarchy - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats instruction hierarchy inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about instruction hierarchy, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what instruction hierarchy must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for instruction hierarchy: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for instruction hierarchy. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Role Framing - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats role framing inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about role framing, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what role framing must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for role framing: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for role framing. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Task Packet - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats task packet inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about task packet, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what task packet must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for task packet: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for task packet. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Constraints - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats constraints inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about constraints, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what constraints must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for constraints: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for constraints. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Examples - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats examples inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about examples, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what examples must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for examples: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for examples. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Evaluation Criteria - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats evaluation criteria inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about evaluation criteria, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what evaluation criteria must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for evaluation criteria: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for evaluation criteria. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Format Contract - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats format contract inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about format contract, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what format contract must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for format contract: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for format contract. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Iterative Prompting - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats iterative prompting inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about iterative prompting, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what iterative prompting must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for iterative prompting: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for iterative prompting. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Debug Prompt - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats debug prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about debug prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what debug prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for debug prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for debug prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Anti-Hallucination Prompt - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats anti-hallucination prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about anti-hallucination prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what anti-hallucination prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for anti-hallucination prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for anti-hallucination prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Research Prompt - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats research prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about research prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what research prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for research prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

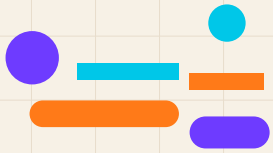
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for research prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Teaching Prompt - Evaluation Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats teaching prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about teaching prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what teaching prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for teaching prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for teaching prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Writing Prompt - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats writing prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about writing prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what writing prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for writing prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for writing prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Coding Prompt - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats coding prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about coding prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what coding prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for coding prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for coding prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Marketing Prompt - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats marketing prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about marketing prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what marketing prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for marketing prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for marketing prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Data Prompt - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats data prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about data prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what data prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for data prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for data prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Decision Prompt - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats decision prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about decision prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what decision prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for decision prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for decision prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Socratic Prompt - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats Socratic prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about Socratic prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what Socratic prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for Socratic prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for Socratic prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Critique Prompt - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats critique prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about critique prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what critique prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for critique prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for critique prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Planning Prompt - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats planning prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about planning prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what planning prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for planning prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for planning prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Translation Prompt - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats translation prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about translation prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what translation prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for translation prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for translation prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Image Prompt - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats image prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about image prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what image prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for image prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for image prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Agent Prompt - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats agent prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about agent prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what agent prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for agent prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for agent prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

System Prompt - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats system prompt inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about system prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what system prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for system prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for system prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Prompt Library - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats prompt library inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about prompt library, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what prompt library must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for prompt library: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for prompt library. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Prompt Qa - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats prompt QA inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about prompt QA, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what prompt QA must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for prompt QA: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for prompt QA. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Prompt Evolution - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats prompt evolution inside Prompt Operating System as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about prompt evolution, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what prompt evolution must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for prompt evolution: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Prompt Operating System without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Prompt Operating System operating assistant for prompt evolution. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Memory Purpose - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats memory purpose inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about memory purpose, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what memory purpose must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for memory purpose: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for memory purpose. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Memory Boundary - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats memory boundary inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about memory boundary, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what memory boundary must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for memory boundary: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for memory boundary. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Project Container - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats project container inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about project container, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what project container must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for project container: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for project container. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Project-Only Memory - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats project-only memory inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about project-only memory, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what project-only memory must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for project-only memory: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

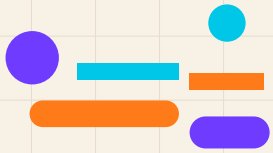
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for project-only memory. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Custom Instructions - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats custom instructions inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about custom instructions, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what custom instructions must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for custom instructions: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

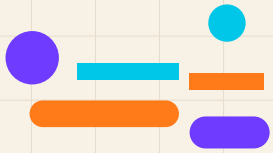
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for custom instructions. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Files As Context - Evaluation Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats files as context inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about files as context, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what files as context must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for files as context: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for files as context. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Conversation History - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats conversation history inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about conversation history, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what conversation history must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for conversation history: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

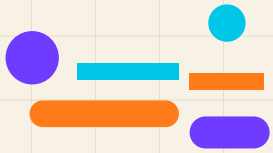
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for conversation history. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Context Compression - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats context compression inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about context compression, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what context compression must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for context compression: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for context compression. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Context Retrieval - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats context retrieval inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about context retrieval, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what context retrieval must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for context retrieval: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for context retrieval. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Context Decay - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats context decay inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about context decay, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what context decay must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for context decay: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for context decay. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Topic Continuity - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats topic continuity inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about topic continuity, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what topic continuity must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for topic continuity: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for topic continuity. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Private Data Hygiene - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats private data hygiene inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about private data hygiene, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what private data hygiene must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for private data hygiene: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

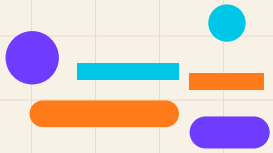
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for private data hygiene. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Workspace Segmentation - Agentic Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats workspace segmentation inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about workspace segmentation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what workspace segmentation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for workspace segmentation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for workspace segmentation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Project Constitution - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats project constitution inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about project constitution, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what project constitution must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for project constitution: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for project constitution. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Team Projects - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats team projects inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about team projects, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what team projects must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for team projects: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for team projects. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Knowledge Folders - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats knowledge folders inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about knowledge folders, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what knowledge folders must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for knowledge folders: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

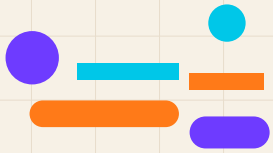
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for knowledge folders. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Memory Audit - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats memory audit inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about memory audit, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what memory audit must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for memory audit: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

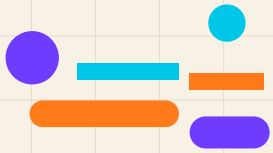
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for memory audit. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Versioned Context - Evaluation Engine



ChatGPT | MCD | OS chunk | Court-tested

OPERATING THESIS

This chunk treats versioned context inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about versioned context, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what versioned context must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for versioned context: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for versioned context. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Context Collision - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats context collision inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about context collision, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what context collision must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for context collision: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for context collision. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Context Refresh - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats context refresh inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about context refresh, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what context refresh must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for context refresh: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for context refresh. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Context Pruning - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats context pruning inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about context pruning, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what context pruning must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for context pruning: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for context pruning. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Handoff Summary - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats handoff summary inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about handoff summary, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what handoff summary must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for handoff summary: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for handoff summary. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

State Repair - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats state repair inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about state repair, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what state repair must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for state repair: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

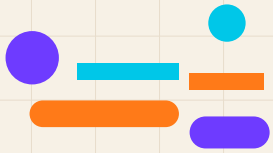
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for state repair. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Personalization Risk - Evaluation Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats personalization risk inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about personalization risk, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what personalization risk must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for personalization risk: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for personalization risk. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Privacy-By-Design - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats privacy-by-design inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about privacy-by-design, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what privacy-by-design must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for privacy-by-design: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for privacy-by-design. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Context Metrics - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats context metrics inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about context metrics, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what context metrics must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for context metrics: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

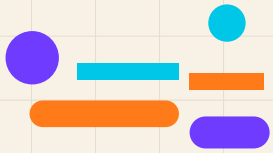
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for context metrics. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Memory Court - Evaluation Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats memory court inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about memory court, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what memory court must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for memory court: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for memory court. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Context Kernel - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats context kernel inside Memory, Projects, Context as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about context kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what context kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for context kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Memory, Projects, Context without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Memory, Projects, Context operating assistant for context kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Research Question - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats research question inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about research question, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what research question must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for research question: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for research question. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Source Selection - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats source selection inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about source selection, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what source selection must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for source selection: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for source selection. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Query Expansion - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats query expansion inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about query expansion, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what query expansion must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for query expansion: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for query expansion. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Literature Scan - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats literature scan inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about literature scan, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what literature scan must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for literature scan: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for literature scan. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Citation Hygiene - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats citation hygiene inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about citation hygiene, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what citation hygiene must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for citation hygiene: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for citation hygiene. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Claim Ladder - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats claim ladder inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about claim ladder, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what claim ladder must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for claim ladder: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for claim ladder. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Evidence Matrix - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats evidence matrix inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about evidence matrix, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what evidence matrix must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for evidence matrix: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for evidence matrix. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Counter-Source Search - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats counter-source search inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about counter-source search, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what counter-source search must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for counter-source search: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for counter-source search. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Synthesis Table - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats synthesis table inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about synthesis table, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what synthesis table must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for synthesis table: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for synthesis table. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Method Summary - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats method summary inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about method summary, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what method summary must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for method summary: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for method summary. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Gap Finding - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats gap finding inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about gap finding, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what gap finding must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for gap finding: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for gap finding. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Hypothesis Generation - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats hypothesis generation inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about hypothesis generation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what hypothesis generation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for hypothesis generation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for hypothesis generation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Research Memo - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats research memo inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about research memo, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what research memo must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for research memo: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for research memo. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Deep Research Planning - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats deep research planning inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about deep research planning, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what deep research planning must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for deep research planning: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for deep research planning. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Documented Report - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats documented report inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about documented report, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what documented report must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for documented report: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for documented report. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Uncertainty Ledger - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats uncertainty ledger inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about uncertainty ledger, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what uncertainty ledger must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for uncertainty ledger: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for uncertainty ledger. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Domain Primer - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats domain primer inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about domain primer, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what domain primer must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for domain primer: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for domain primer. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Expert Interview Prep - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats expert interview prep inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about expert interview prep, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what expert interview prep must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for expert interview prep: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for expert interview prep. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Paper Outline - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats paper outline inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about paper outline, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what paper outline must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for paper outline: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for paper outline. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Survey Article - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats survey article inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about survey article, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what survey article must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for survey article: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for survey article. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Argument Map - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats argument map inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about argument map, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what argument map must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for argument map: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for argument map. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Fact Check - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats fact check inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about fact check, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what fact check must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for fact check: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for fact check. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Replication Mindset - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats replication mindset inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about replication mindset, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what replication mindset must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for replication mindset: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for replication mindset. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Citation Limits - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats citation limits inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about citation limits, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what citation limits must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for citation limits: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for citation limits. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Research Ethics - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats research ethics inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about research ethics, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what research ethics must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for research ethics: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

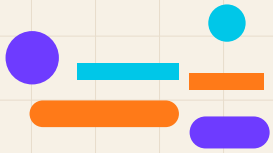
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for research ethics. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Research Assistant Loop - Evaluation Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats research assistant loop inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about research assistant loop, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what research assistant loop must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for research assistant loop: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for research assistant loop. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Research Qa - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats research QA inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about research QA, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what research QA must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for research QA: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for research QA. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Research Kernel - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats research kernel inside Research & Deep Work as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about research kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what research kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for research kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Research & Deep Work without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Research & Deep Work operating assistant for research kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Draft Workspace - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats draft workspace inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about draft workspace, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what draft workspace must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for draft workspace: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for draft workspace. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Inline Revision - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats inline revision inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about inline revision, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what inline revision must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for inline revision: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for inline revision. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Section Focus - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats section focus inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about section focus, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what section focus must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for section focus: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for section focus. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Style Pass - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats style pass inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about style pass, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what style pass must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for style pass: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for style pass. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Argument Pass - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats argument pass inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about argument pass, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what argument pass must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for argument pass: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for argument pass. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Structure Pass - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats structure pass inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about structure pass, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what structure pass must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for structure pass: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for structure pass. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Tone Pass - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats tone pass inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about tone pass, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what tone pass must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for tone pass: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for tone pass. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Rewrite Ethics - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats rewrite ethics inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about rewrite ethics, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what rewrite ethics must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for rewrite ethics: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for rewrite ethics. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Chapter Architecture - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats chapter architecture inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about chapter architecture, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what chapter architecture must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for chapter architecture: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for chapter architecture. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Book Outline - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats book outline inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about book outline, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what book outline must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for book outline: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for book outline. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Editing Protocol - Agentic Engine

ChatGPT | MCD | OS chunk | Court-tested

OPERATING THESIS

This chunk treats editing protocol inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about editing protocol, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what editing protocol must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for editing protocol: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for editing protocol. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Copy Improvement - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats copy improvement inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about copy improvement, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what copy improvement must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for copy improvement: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for copy improvement. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Code Revision - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats code revision inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about code revision, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what code revision must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for code revision: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for code revision. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Comment Handling - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats comment handling inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about comment handling, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what comment handling must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for comment handling: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for comment handling. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Version Comparison - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats version comparison inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about version comparison, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what version comparison must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for version comparison: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for version comparison. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Reader Persona - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats reader persona inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about reader persona, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what reader persona must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for reader persona: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for reader persona. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Pedagogy Pass - Agentic Engine

ChatGPT | MCD | OS chunk | Court-tested

OPERATING THESIS

This chunk treats pedagogy pass inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about pedagogy pass, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what pedagogy pass must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for pedagogy pass: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for pedagogy pass. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Executive Summary - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats executive summary inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about executive summary, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what executive summary must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for executive summary: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for executive summary. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Proposal Draft - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats proposal draft inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about proposal draft, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what proposal draft must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for proposal draft: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for proposal draft. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Blog Draft - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats blog draft inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about blog draft, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what blog draft must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for blog draft: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for blog draft. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Ebook Chapter - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats ebook chapter inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about ebook chapter, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what ebook chapter must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for ebook chapter: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for ebook chapter. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Script Draft - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats script draft inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about script draft, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what script draft must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for script draft: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for script draft. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Email Draft - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats email draft inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about email draft, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what email draft must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for email draft: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for email draft. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Policy Draft - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats policy draft inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about policy draft, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what policy draft must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for policy draft: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for policy draft. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Peer Review - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats peer review inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about peer review, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what peer review must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for peer review: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for peer review. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Canvas Qa - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats canvas QA inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about canvas QA, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what canvas QA must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for canvas QA: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for canvas QA. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Publication Pass - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats publication pass inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about publication pass, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what publication pass must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for publication pass: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for publication pass. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Writing Kernel - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats writing kernel inside Canvas & Writing Workbench as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about writing kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what writing kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for writing kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Canvas & Writing Workbench without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Canvas & Writing Workbench operating assistant for writing kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

File Ingestion - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats file ingestion inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about file ingestion, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what file ingestion must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for file ingestion: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for file ingestion. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Spreadsheet Inspection - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats spreadsheet inspection inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about spreadsheet inspection, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what spreadsheet inspection must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for spreadsheet inspection: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for spreadsheet inspection. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Table Cleaning - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats table cleaning inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about table cleaning, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what table cleaning must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for table cleaning: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for table cleaning. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Statistical Caution - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats statistical caution inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about statistical caution, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what statistical caution must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for statistical caution: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for statistical caution. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Chart Generation - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats chart generation inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about chart generation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what chart generation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for chart generation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for chart generation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Data Questions - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats data questions inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about data questions, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what data questions must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for data questions: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for data questions. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Missing Values - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats missing values inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about missing values, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what missing values must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for missing values: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for missing values. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Outlier Thinking - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats outlier thinking inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about outlier thinking, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what outlier thinking must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for outlier thinking: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for outlier thinking. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Feature Engineering - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats feature engineering inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about feature engineering, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what feature engineering must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for feature engineering: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for feature engineering. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Summary Tables - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats summary tables inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about summary tables, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what summary tables must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for summary tables: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

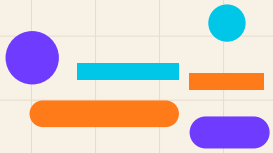
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for summary tables. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Notebook Reasoning - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats notebook reasoning inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about notebook reasoning, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what notebook reasoning must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for notebook reasoning: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for notebook reasoning. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Business Dashboard - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats business dashboard inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about business dashboard, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what business dashboard must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for business dashboard: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

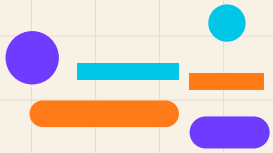
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for business dashboard. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Experimental Data - Agentic Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats experimental data inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about experimental data, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what experimental data must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for experimental data: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for experimental data. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Survey Results - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats survey results inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about survey results, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what survey results must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for survey results: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for survey results. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Pdf Extraction - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats PDF extraction inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about PDF extraction, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what PDF extraction must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for PDF extraction: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for PDF extraction. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Csv Workflow - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats CSV workflow inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about CSV workflow, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what CSV workflow must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for CSV workflow: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for CSV workflow. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Data Provenance - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats data provenance inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about data provenance, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what data provenance must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for data provenance: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for data provenance. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Code Transparency - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats code transparency inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about code transparency, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what code transparency must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for code transparency: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for code transparency. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Calculation Verification - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats calculation verification inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about calculation verification, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what calculation verification must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for calculation verification: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

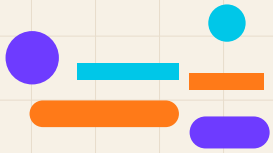
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for calculation verification. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Visualization Limits - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats visualization limits inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about visualization limits, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what visualization limits must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for visualization limits: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for visualization limits. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Privacy In Files - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats privacy in files inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about privacy in files, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what privacy in files must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for privacy in files: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for privacy in files. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Reproducibility - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats reproducibility inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about reproducibility, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what reproducibility must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for reproducibility: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for reproducibility. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Data Narrative - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats data narrative inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about data narrative, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what data narrative must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for data narrative: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

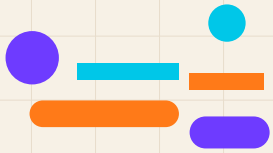
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for data narrative. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Model Evaluation - Evaluation Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats model evaluation inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about model evaluation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what model evaluation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for model evaluation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for model evaluation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Analytics Memo - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats analytics memo inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about analytics memo, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what analytics memo must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for analytics memo: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for analytics memo. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Data Qa - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats data QA inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about data QA, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what data QA must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for data QA: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for data QA. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

File Retention Awareness - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats file retention awareness inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about file retention awareness, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what file retention awareness must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for file retention awareness: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for file retention awareness. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Analysis Kernel - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats analysis kernel inside Data Analysis & Files as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about analysis kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what analysis kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for analysis kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Data Analysis & Files without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Data Analysis & Files operating assistant for analysis kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Freshness Trigger - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats freshness trigger inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about freshness trigger, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what freshness trigger must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for freshness trigger: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for freshness trigger. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Search Decision - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats search decision inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about search decision, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what search decision must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for search decision: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for search decision. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Source Triage - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats source triage inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about source triage, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what source triage must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for source triage: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for source triage. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

News Caution - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats news caution inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about news caution, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what news caution must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for news caution: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for news caution. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Local Info - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats local info inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about local info, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what local info must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for local info: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for local info. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Pricing Checks - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats pricing checks inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about pricing checks, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what pricing checks must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for pricing checks: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for pricing checks. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Law Checks - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats law checks inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about law checks, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what law checks must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for law checks: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for law checks. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Medical Caution - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats medical caution inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about medical caution, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what medical caution must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for medical caution: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for medical caution. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Finance Caution - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats finance caution inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about finance caution, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what finance caution must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for finance caution: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for finance caution. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Release Notes - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats release notes inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about release notes, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what release notes must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for release notes: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for release notes. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Product Docs - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats product docs inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about product docs, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what product docs must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for product docs: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for product docs. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Source Comparison - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats source comparison inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about source comparison, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what source comparison must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for source comparison: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for source comparison. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Search Query Design - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats search query design inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about search query design, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what search query design must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for search query design: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for search query design. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Source Recency - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats source recency inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about source recency, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what source recency must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for source recency: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for source recency. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Source Authority - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats source authority inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about source authority, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what source authority must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for source authority: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for source authority. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Citation Discipline - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats citation discipline inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about citation discipline, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what citation discipline must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for citation discipline: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for citation discipline. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Fact Versus Interpretation - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats fact versus interpretation inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about fact versus interpretation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what fact versus interpretation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for fact versus interpretation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for fact versus interpretation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Changed Fact - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats changed fact inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about changed fact, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what changed fact must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for changed fact: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for changed fact. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Crawl Limitation - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats crawl limitation inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about crawl limitation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what crawl limitation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for crawl limitation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for crawl limitation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Broken Sources - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats broken sources inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about broken sources, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what broken sources must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for broken sources: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for broken sources. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Link Evaluation - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats link evaluation inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about link evaluation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what link evaluation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for link evaluation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for link evaluation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Timely Answer - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats timely answer inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about timely answer, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what timely answer must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for timely answer: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for timely answer. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Web Result Ranking - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats web result ranking inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about web result ranking, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what web result ranking must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for web result ranking: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for web result ranking. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Search Hallucination - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats search hallucination inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about search hallucination, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what search hallucination must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for search hallucination: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for search hallucination. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Source Quote Limit - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats source quote limit inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about source quote limit, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what source quote limit must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for source quote limit: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for source quote limit. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Search Qa - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats search QA inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about search QA, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what search QA must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for search QA: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for search QA. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Update Cadence - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats update cadence inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about update cadence, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what update cadence must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for update cadence: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for update cadence. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Freshness Kernel - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats freshness kernel inside Web Search & Freshness as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about freshness kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what freshness kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for freshness kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Web Search & Freshness without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Web Search & Freshness operating assistant for freshness kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Image Input - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats image input inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about image input, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what image input must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for image input: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for image input. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Visual Description - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats visual description inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about visual description, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what visual description must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for visual description: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for visual description. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Diagram Reading - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats diagram reading inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about diagram reading, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what diagram reading must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for diagram reading: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for diagram reading. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Photo Limits - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats photo limits inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about photo limits, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what photo limits must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for photo limits: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for photo limits. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Identity Boundary - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats identity boundary inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about identity boundary, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what identity boundary must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for identity boundary: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for identity boundary. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Image Generation - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats image generation inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about image generation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what image generation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for image generation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for image generation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Style Prompt - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats style prompt inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about style prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what style prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for style prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for style prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Brand Consistency - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats brand consistency inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about brand consistency, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what brand consistency must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for brand consistency: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for brand consistency. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Thumbnail Test - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats thumbnail test inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about thumbnail test, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what thumbnail test must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for thumbnail test: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for thumbnail test. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Accessibility Text - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats accessibility text inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about accessibility text, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what accessibility text must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for accessibility text: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for accessibility text. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Voice Conversation - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats voice conversation inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about voice conversation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what voice conversation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for voice conversation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for voice conversation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Meeting Notes - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats meeting notes inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about meeting notes, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what meeting notes must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for meeting notes: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for meeting notes. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Recording Caution - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats recording caution inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about recording caution, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what recording caution must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for recording caution: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for recording caution. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Multimodal Context - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats multimodal context inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about multimodal context, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what multimodal context must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for multimodal context: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for multimodal context. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Visual Reasoning - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats visual reasoning inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about visual reasoning, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what visual reasoning must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for visual reasoning: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for visual reasoning. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Chart Interpretation - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats chart interpretation inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about chart interpretation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what chart interpretation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for chart interpretation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for chart interpretation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Screenshot Workflow - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats screenshot workflow inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about screenshot workflow, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what screenshot workflow must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for screenshot workflow: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for screenshot workflow. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Ui Inspection - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats UI inspection inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about UI inspection, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what UI inspection must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for UI inspection: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for UI inspection. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Creative Direction - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats creative direction inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about creative direction, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what creative direction must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for creative direction: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for creative direction. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Prompt-To-Image - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats prompt-to-image inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about prompt-to-image, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what prompt-to-image must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for prompt-to-image: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for prompt-to-image. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Image Qa - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats image QA inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about image QA, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what image QA must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for image QA: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for image QA. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Visual Ethics - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats visual ethics inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about visual ethics, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what visual ethics must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for visual ethics: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for visual ethics. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Copyright Caution - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats copyright caution inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about copyright caution, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what copyright caution must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for copyright caution: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for copyright caution. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Brand-Safe Imagery - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats brand-safe imagery inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about brand-safe imagery, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what brand-safe imagery must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for brand-safe imagery: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for brand-safe imagery. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Alt Text - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats alt text inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about alt text, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what alt text must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for alt text: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for alt text. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Presentation Visuals - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats presentation visuals inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about presentation visuals, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what presentation visuals must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for presentation visuals: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for presentation visuals. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Multimodal Court - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats multimodal court inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about multimodal court, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what multimodal court must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for multimodal court: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for multimodal court. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Visual Kernel - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats visual kernel inside Image, Multimodal, Voice as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about visual kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what visual kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for visual kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Image, Multimodal, Voice without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Image, Multimodal, Voice operating assistant for visual kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Code Explanation - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats code explanation inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about code explanation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what code explanation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for code explanation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for code explanation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Debugging Loop - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats debugging loop inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about debugging loop, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what debugging loop must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for debugging loop: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for debugging loop. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Test Generation - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats test generation inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about test generation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what test generation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for test generation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for test generation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Architecture Review - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats architecture review inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about architecture review, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what architecture review must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for architecture review: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for architecture review. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Api Design - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats API design inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about API design, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what API design must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for API design: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

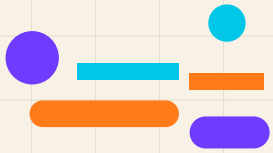
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for API design. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Security Review - Evaluation Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats security review inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about security review, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what security review must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for security review: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for security review. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Refactor Plan - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats refactor plan inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about refactor plan, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what refactor plan must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for refactor plan: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for refactor plan. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Sql Query - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats SQL query inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about SQL query, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what SQL query must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for SQL query: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for SQL query. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Regex Help - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats regex help inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about regex help, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what regex help must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for regex help: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for regex help. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Devops Prompt - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats devops prompt inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about devops prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what devops prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for devops prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for devops prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Logs Analysis - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats logs analysis inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about logs analysis, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what logs analysis must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for logs analysis: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for logs analysis. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Documentation - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats documentation inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about documentation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what documentation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for documentation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for documentation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Pull-Request Review - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats pull-request review inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about pull-request review, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what pull-request review must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for pull-request review: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for pull-request review. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Algorithm Sketch - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats algorithm sketch inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about algorithm sketch, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what algorithm sketch must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for algorithm sketch: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for algorithm sketch. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Performance Thinking - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats performance thinking inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about performance thinking, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what performance thinking must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for performance thinking: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for performance thinking. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Error Reproduction - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats error reproduction inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about error reproduction, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what error reproduction must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for error reproduction: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for error reproduction. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Package Drift - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats package drift inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about package drift, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what package drift must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for package drift: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for package drift. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Version Specificity - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats version specificity inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about version specificity, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what version specificity must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for version specificity: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for version specificity. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Sandbox Limits - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats sandbox limits inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about sandbox limits, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what sandbox limits must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for sandbox limits: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for sandbox limits. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Secret Hygiene - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats secret hygiene inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about secret hygiene, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what secret hygiene must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for secret hygiene: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for secret hygiene. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Code Diff - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats code diff inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about code diff, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what code diff must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for code diff: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for code diff. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Pseudocode - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats pseudocode inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about pseudocode, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what pseudocode must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for pseudocode: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for pseudocode. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Learning Code - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats learning code inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about learning code, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what learning code must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for learning code: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for learning code. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Automation Script - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats automation script inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about automation script, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what automation script must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for automation script: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for automation script. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Prompt-To-Code - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats prompt-to-code inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about prompt-to-code, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what prompt-to-code must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for prompt-to-code: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for prompt-to-code. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Code Qa - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats code QA inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about code QA, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what code QA must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for code QA: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for code QA. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Engineering Court - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats engineering court inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about engineering court, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what engineering court must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for engineering court: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for engineering court. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Coding Kernel - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats coding kernel inside Coding & Engineering as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about coding kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what coding kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for coding kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Coding & Engineering without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Coding & Engineering operating assistant for coding kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Strategy Memo - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats strategy memo inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about strategy memo, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what strategy memo must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for strategy memo: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for strategy memo. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Market Research - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats market research inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about market research, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what market research must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for market research: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for market research. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Customer Persona - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats customer persona inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about customer persona, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what customer persona must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for customer persona: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for customer persona. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Sales Script - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats sales script inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about sales script, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what sales script must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for sales script: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for sales script. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Support Triage - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats support triage inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about support triage, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what support triage must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for support triage: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for support triage. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Proposal Engine - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats proposal engine inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about proposal engine, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what proposal engine must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for proposal engine: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

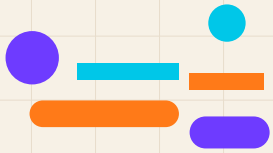
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for proposal engine. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Sop Builder - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats SOP builder inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about SOP builder, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what SOP builder must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for SOP builder: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for SOP builder. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Meeting Prep - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats meeting prep inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about meeting prep, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what meeting prep must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for meeting prep: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for meeting prep. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Meeting Recap - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats meeting recap inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about meeting recap, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what meeting recap must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for meeting recap: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

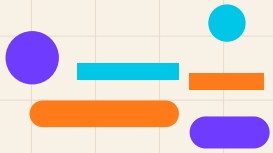
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for meeting recap. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Competitive Analysis - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats competitive analysis inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about competitive analysis, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what competitive analysis must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for competitive analysis: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for competitive analysis. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Pricing Draft - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats pricing draft inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about pricing draft, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what pricing draft must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for pricing draft: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for pricing draft. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Risk Register - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats risk register inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about risk register, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what risk register must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for risk register: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for risk register. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Okr Planning - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats OKR planning inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about OKR planning, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what OKR planning must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for OKR planning: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for OKR planning. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Project Plan - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats project plan inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about project plan, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what project plan must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for project plan: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for project plan. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Procurement Comparison - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats procurement comparison inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about procurement comparison, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what procurement comparison must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for procurement comparison: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

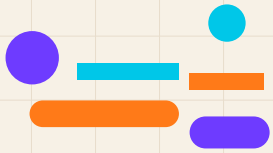
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for procurement comparison. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Vendor Evaluation - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats vendor evaluation inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about vendor evaluation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what vendor evaluation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for vendor evaluation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for vendor evaluation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Crm Notes - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats CRM notes inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about CRM notes, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what CRM notes must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for CRM notes: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for CRM notes. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Training Plan - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats training plan inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about training plan, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what training plan must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for training plan: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

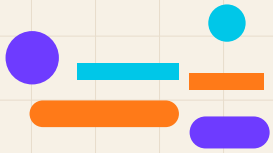
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for training plan. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Policy Summarization - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats policy summarization inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about policy summarization, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what policy summarization must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for policy summarization: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

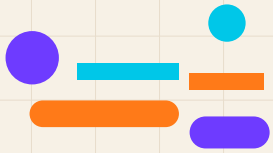
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for policy summarization. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Hr Communication - Evaluation Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats HR communication inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about HR communication, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what HR communication must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for HR communication: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for HR communication. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Finance Narrative - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats finance narrative inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about finance narrative, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what finance narrative must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for finance narrative: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for finance narrative. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Operations Audit - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats operations audit inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about operations audit, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what operations audit must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for operations audit: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

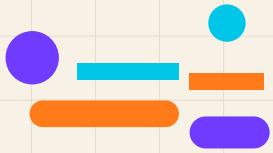
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for operations audit. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Customer Journey - Evaluation Engine



ChatGPT | MCD | OS chunk | Court-tested

OPERATING THESIS

This chunk treats customer journey inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about customer journey, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what customer journey must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for customer journey: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for customer journey. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Go-To-Market - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats go-to-market inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about go-to-market, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what go-to-market must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for go-to-market: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

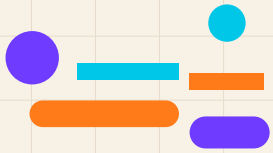
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for go-to-market. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Business Metrics - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats business metrics inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about business metrics, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what business metrics must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for business metrics: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

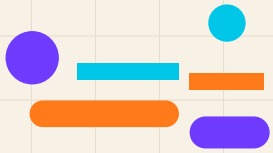
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for business metrics. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Workflow Qa - Evaluation Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats workflow QA inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about workflow QA, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what workflow QA must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for workflow QA: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for workflow QA. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Adoption Court - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats adoption court inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about adoption court, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what adoption court must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for adoption court: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for adoption court. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Business Kernel - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats business kernel inside Business Workflows as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about business kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what business kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for business kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Business Workflows without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Business Workflows operating assistant for business kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

101 Explanation - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats 101 explanation inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about 101 explanation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what 101 explanation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for 101 explanation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for 101 explanation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

For-Dummies Style - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats for-dummies style inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about for-dummies style, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what for-dummies style must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for for-dummies style: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

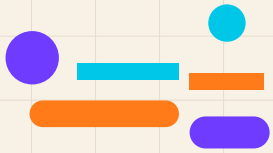
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for for-dummies style. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Diagnostic Quiz - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats diagnostic quiz inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about diagnostic quiz, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what diagnostic quiz must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for diagnostic quiz: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for diagnostic quiz. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Spaced Repetition - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats spaced repetition inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about spaced repetition, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what spaced repetition must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for spaced repetition: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for spaced repetition. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Rubric Design - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats rubric design inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about rubric design, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what rubric design must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for rubric design: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for rubric design. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Lesson Plan - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats lesson plan inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about lesson plan, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what lesson plan must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for lesson plan: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for lesson plan. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Concept Ladder - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats concept ladder inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about concept ladder, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what concept ladder must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for concept ladder: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for concept ladder. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Analogy Use - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats analogy use inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about analogy use, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what analogy use must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for analogy use: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for analogy use. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Worked Example - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats worked example inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about worked example, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what worked example must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for worked example: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for worked example. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Misconception Repair - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats misconception repair inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about misconception repair, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what misconception repair must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for misconception repair: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for misconception repair. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Socratic Tutor - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats Socratic tutor inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about Socratic tutor, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what Socratic tutor must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for Socratic tutor: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for Socratic tutor. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Exam Prep - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats exam prep inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about exam prep, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what exam prep must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for exam prep: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for exam prep. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Skill Drill - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats skill drill inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about skill drill, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what skill drill must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for skill drill: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for skill drill. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Study Schedule - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats study schedule inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about study schedule, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what study schedule must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for study schedule: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for study schedule. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Feedback Loop - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats feedback loop inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about feedback loop, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what feedback loop must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for feedback loop: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for feedback loop. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Learning Objective - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats learning objective inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about learning objective, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what learning objective must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for learning objective: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for learning objective. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Curriculum Map - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats curriculum map inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about curriculum map, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what curriculum map must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for curriculum map: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for curriculum map. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Teaching Persona - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats teaching persona inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about teaching persona, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what teaching persona must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for teaching persona: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for teaching persona. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Knowledge Compression - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats knowledge compression inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about knowledge compression, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what knowledge compression must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for knowledge compression: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for knowledge compression. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Assessment Design - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats assessment design inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about assessment design, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what assessment design must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for assessment design: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for assessment design. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Beginner To Expert - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats beginner to expert inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about beginner to expert, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what beginner to expert must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for beginner to expert: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for beginner to expert. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Translation For Learners - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats translation for learners inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about translation for learners, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what translation for learners must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for translation for learners: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for translation for learners. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Pedagogical Tone - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats pedagogical tone inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about pedagogical tone, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what pedagogical tone must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for pedagogical tone: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for pedagogical tone. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Cognitive Load - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats cognitive load inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about cognitive load, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what cognitive load must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for cognitive load: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for cognitive load. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Student Safety - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats student safety inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about student safety, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what student safety must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for student safety: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for student safety. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Learning Qa - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats learning QA inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about learning QA, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what learning QA must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for learning QA: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for learning QA. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Mastery Court - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats mastery court inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about mastery court, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what mastery court must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for mastery court: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for mastery court. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Pedagogy Kernel - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats pedagogy kernel inside Learning & Pedagogy as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about pedagogy kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what pedagogy kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for pedagogy kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Learning & Pedagogy without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Learning & Pedagogy operating assistant for pedagogy kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Ideation - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats ideation inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about ideation, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what ideation must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for ideation: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for ideation. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Concept Art Brief - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats concept art brief inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about concept art brief, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what concept art brief must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for concept art brief: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for concept art brief. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Brand Voice - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats brand voice inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about brand voice, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what brand voice must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for brand voice: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for brand voice. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Story Engine - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats story engine inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about story engine, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what story engine must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for story engine: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for story engine. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Script Writing - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats script writing inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about script writing, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what script writing must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for script writing: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for script writing. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Video Outline - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats video outline inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about video outline, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what video outline must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for video outline: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for video outline. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Podcast Plan - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats podcast plan inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about podcast plan, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what podcast plan must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for podcast plan: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for podcast plan. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Social Posts - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats social posts inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about social posts, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what social posts must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for social posts: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for social posts. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Meme Caution - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats meme caution inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about meme caution, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what meme caution must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for meme caution: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for meme caution. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Thumbnail Conversion - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats thumbnail conversion inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about thumbnail conversion, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what thumbnail conversion must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for thumbnail conversion: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for thumbnail conversion. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Hook Design - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats hook design inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about hook design, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what hook design must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for hook design: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for hook design. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Copywriting - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats copywriting inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about copywriting, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what copywriting must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for copywriting: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for copywriting. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Ugc Concept - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats UGC concept inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about UGC concept, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what UGC concept must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for UGC concept: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for UGC concept. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Ad Variants - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats ad variants inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about ad variants, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what ad variants must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for ad variants: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for ad variants. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Creative Constraints - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats creative constraints inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about creative constraints, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what creative constraints must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for creative constraints: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for creative constraints. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Style Board - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats style board inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about style board, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what style board must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for style board: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for style board. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Music Project - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats music project inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about music project, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what music project must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for music project: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for music project. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Narrative Arc - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats narrative arc inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about narrative arc, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what narrative arc must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for narrative arc: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

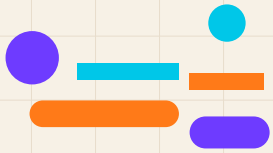
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for narrative arc. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Campaign Assets - Agentic Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats campaign assets inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about campaign assets, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what campaign assets must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for campaign assets: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

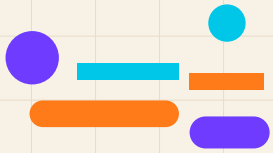
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for campaign assets. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Persona Consistency - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats persona consistency inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about persona consistency, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what persona consistency must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for persona consistency: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for persona consistency. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Taste Calibration - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats taste calibration inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about taste calibration, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what taste calibration must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for taste calibration: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for taste calibration. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Creative Qa - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats creative QA inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about creative QA, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what creative QA must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for creative QA: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for creative QA. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

False Novelty - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats false novelty inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about false novelty, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what false novelty must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for false novelty: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for false novelty. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Originality Check - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats originality check inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about originality check, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what originality check must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for originality check: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for originality check. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Media Ethics - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats media ethics inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about media ethics, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what media ethics must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for media ethics: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for media ethics. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Production Prompt - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats production prompt inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about production prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what production prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for production prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for production prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Creative Court - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats creative court inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about creative court, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what creative court must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for creative court: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for creative court. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Media Kernel - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats media kernel inside Creativity & Media as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about media kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what media kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for media kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Creativity & Media without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Creativity & Media operating assistant for media kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Daily Planning - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats daily planning inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about daily planning, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what daily planning must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for daily planning: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for daily planning. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Task Triage - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats task triage inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about task triage, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what task triage must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for task triage: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for task triage. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Calendar Thinking - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats calendar thinking inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about calendar thinking, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what calendar thinking must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for calendar thinking: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for calendar thinking. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Email Drafting - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats email drafting inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about email drafting, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what email drafting must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for email drafting: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for email drafting. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Decision Journal - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats decision journal inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about decision journal, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what decision journal must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for decision journal: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for decision journal. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Habit Loop - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats habit loop inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about habit loop, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what habit loop must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for habit loop: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for habit loop. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Reflection - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats reflection inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about reflection, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what reflection must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for reflection: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for reflection. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Focus Mode - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats focus mode inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about focus mode, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what focus mode must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for focus mode: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for focus mode. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Overload Reduction - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats overload reduction inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about overload reduction, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what overload reduction must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for overload reduction: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for overload reduction. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Project Next Action - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats project next action inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about project next action, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what project next action must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for project next action: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for project next action. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Personal Crm - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats personal CRM inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about personal CRM, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what personal CRM must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for personal CRM: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for personal CRM. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Travel Planning - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats travel planning inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about travel planning, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what travel planning must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for travel planning: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for travel planning. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Shopping Research - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats shopping research inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about shopping research, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what shopping research must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for shopping research: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for shopping research. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Health Caution - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats health caution inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about health caution, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what health caution must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for health caution: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for health caution. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Legal Caution - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats legal caution inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about legal caution, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what legal caution must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for legal caution: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for legal caution. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Financial Caution - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats financial caution inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about financial caution, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what financial caution must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for financial caution: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for financial caution. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Motivation Support - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats motivation support inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about motivation support, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what motivation support must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for motivation support: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for motivation support. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Routine Builder - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats routine builder inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about routine builder, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what routine builder must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for routine builder: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for routine builder. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Goal Decomposition - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats goal decomposition inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about goal decomposition, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what goal decomposition must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for goal decomposition: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for goal decomposition. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Review Ritual - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats review ritual inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about review ritual, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what review ritual must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for review ritual: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for review ritual. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Attention Management - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats attention management inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about attention management, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what attention management must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for attention management: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for attention management. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Energy Planning - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats energy planning inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about energy planning, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what energy planning must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for energy planning: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for energy planning. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Friction Removal - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats friction removal inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about friction removal, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what friction removal must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for friction removal: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for friction removal. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Personal Knowledge Base - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats personal knowledge base inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about personal knowledge base, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what personal knowledge base must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for personal knowledge base: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for personal knowledge base. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Automation Idea - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats automation idea inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about automation idea, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what automation idea must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for automation idea: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for automation idea. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Productivity Qa - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats productivity QA inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about productivity QA, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what productivity QA must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for productivity QA: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for productivity QA. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Dependence Court - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats dependence court inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about dependence court, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what dependence court must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for dependence court: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for dependence court. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Productivity Kernel - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats productivity kernel inside Personal Productivity as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about productivity kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what productivity kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for productivity kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Personal Productivity without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Personal Productivity operating assistant for productivity kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Workspace Rollout - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats workspace rollout inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about workspace rollout, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what workspace rollout must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for workspace rollout: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for workspace rollout. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Policy Baseline - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats policy baseline inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about policy baseline, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what policy baseline must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for policy baseline: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for policy baseline. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Data Classification - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats data classification inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about data classification, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what data classification must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for data classification: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for data classification. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Role-Based Usage - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats role-based usage inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about role-based usage, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what role-based usage must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for role-based usage: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for role-based usage. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Prompt Standards - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats prompt standards inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about prompt standards, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what prompt standards must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for prompt standards: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for prompt standards. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Shared Project - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats shared project inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about shared project, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what shared project must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for shared project: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for shared project. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Connector Governance - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats connector governance inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about connector governance, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what connector governance must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for connector governance: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for connector governance. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Approval Workflow - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats approval workflow inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about approval workflow, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what approval workflow must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for approval workflow: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

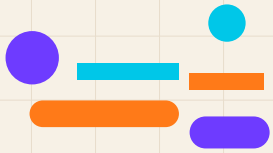
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for approval workflow. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Training Program - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats training program inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about training program, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what training program must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for training program: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for training program. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Champion Network - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats champion network inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about champion network, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what champion network must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for champion network: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for champion network. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Risk Taxonomy - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats risk taxonomy inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about risk taxonomy, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what risk taxonomy must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for risk taxonomy: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for risk taxonomy. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Usage Analytics - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats usage analytics inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about usage analytics, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what usage analytics must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for usage analytics: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for usage analytics. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Evaluation Suite - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats evaluation suite inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about evaluation suite, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what evaluation suite must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for evaluation suite: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for evaluation suite. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Quality Gates - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats quality gates inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about quality gates, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what quality gates must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for quality gates: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for quality gates. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Vendor Review - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats vendor review inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about vendor review, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what vendor review must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for vendor review: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for vendor review. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Compliance Prompt - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats compliance prompt inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about compliance prompt, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what compliance prompt must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for compliance prompt: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for compliance prompt. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Employee Enablement - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats employee enablement inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about employee enablement, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what employee enablement must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for employee enablement: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for employee enablement. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Team Memory - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats team memory inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about team memory, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what team memory must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for team memory: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for team memory. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Shared Assets - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats shared assets inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about shared assets, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what shared assets must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for shared assets: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for shared assets. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Change Management - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats change management inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about change management, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what change management must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for change management: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for change management. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Incident Response - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats incident response inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about incident response, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what incident response must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for incident response: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for incident response. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Model Update Governance - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats model update governance inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about model update governance, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what model update governance must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for model update governance: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for model update governance. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Ai Council - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats AI council inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about AI council, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what AI council must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for AI council: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for AI council. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Adoption Metrics - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats adoption metrics inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about adoption metrics, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what adoption metrics must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for adoption metrics: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for adoption metrics. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Trust Building - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats trust building inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about trust building, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what trust building must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for trust building: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for trust building. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Governance Qa - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats governance QA inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about governance QA, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what governance QA must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for governance QA: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for governance QA. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Organizational Court - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats organizational court inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about organizational court, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what organizational court must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for organizational court: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for organizational court. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Governance Kernel - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats governance kernel inside Team Adoption & Governance as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about governance kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what governance kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for governance kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Team Adoption & Governance without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Team Adoption & Governance operating assistant for governance kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Temporary Chat - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats temporary chat inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about temporary chat, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what temporary chat must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for temporary chat: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for temporary chat. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Data Control - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats data control inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about data control, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what data control must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for data control: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for data control. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Memory Control - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats memory control inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about memory control, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what memory control must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for memory control: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for memory control. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Enterprise Privacy - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats enterprise privacy inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about enterprise privacy, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what enterprise privacy must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for enterprise privacy: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for enterprise privacy. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Sensitive Data - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats sensitive data inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about sensitive data, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what sensitive data must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for sensitive data: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for sensitive data. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Pii Minimization - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats PII minimization inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about PII minimization, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what PII minimization must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for PII minimization: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for PII minimization. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Consent Boundary - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats consent boundary inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about consent boundary, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what consent boundary must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for consent boundary: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for consent boundary. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Medical Safety - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats medical safety inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about medical safety, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what medical safety must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for medical safety: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for medical safety. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Legal Safety - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats legal safety inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about legal safety, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what legal safety must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for legal safety: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for legal safety. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Financial Safety - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats financial safety inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about financial safety, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what financial safety must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for financial safety: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for financial safety. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Child Safety - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats child safety inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about child safety, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what child safety must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for child safety: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for child safety. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Copyright Safety - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats copyright safety inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about copyright safety, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what copyright safety must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for copyright safety: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for copyright safety. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Confidential Data - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats confidential data inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about confidential data, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what confidential data must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for confidential data: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for confidential data. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Prompt Injection - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats prompt injection inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about prompt injection, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what prompt injection must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for prompt injection: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for prompt injection. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Malicious Content - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats malicious content inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about malicious content, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what malicious content must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for malicious content: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for malicious content. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Deception Risk - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats deception risk inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about deception risk, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what deception risk must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for deception risk: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for deception risk. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Overtrust - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats overtrust inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about overtrust, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what overtrust must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for overtrust: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for overtrust. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Bias Check - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats bias check inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about bias check, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what bias check must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for bias check: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for bias check. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Security Controls - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats security controls inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about security controls, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what security controls must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for security controls: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for security controls. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Retention Awareness - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats retention awareness inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about retention awareness, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what retention awareness must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for retention awareness: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for retention awareness. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Export/Delete Thinking - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats export/delete thinking inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about export/delete thinking, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what export/delete thinking must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for export/delete thinking: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for export/delete thinking. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Auditable Workflow - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats auditable workflow inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about auditable workflow, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what auditable workflow must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for auditable workflow: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for auditable workflow. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Risk Disclaimers - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats risk disclaimers inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about risk disclaimers, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what risk disclaimers must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for risk disclaimers: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for risk disclaimers. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Safe Refusal - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats safe refusal inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about safe refusal, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what safe refusal must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for safe refusal: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for safe refusal. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Privacy Ux - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats privacy UX inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about privacy UX, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what privacy UX must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for privacy UX: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for privacy UX. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Safety Qa - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats safety QA inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about safety QA, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what safety QA must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for safety QA: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for safety QA. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Hallucination Court - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats hallucination court inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about hallucination court, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what hallucination court must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for hallucination court: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for hallucination court. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Safety Kernel - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats safety kernel inside Privacy, Safety & Data Controls as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about safety kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what safety kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for safety kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Privacy, Safety & Data Controls without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Privacy, Safety & Data Controls operating assistant for safety kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Agent Mode - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats agent mode inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about agent mode, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what agent mode must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for agent mode: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for agent mode. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Tool Choosing - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats tool choosing inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about tool choosing, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what tool choosing must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for tool choosing: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for tool choosing. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Task Decomposition - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats task decomposition inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about task decomposition, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what task decomposition must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for task decomposition: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for task decomposition. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Action Approval - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats action approval inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about action approval, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what action approval must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for action approval: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for action approval. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Browser Tasks - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats browser tasks inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about browser tasks, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what browser tasks must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for browser tasks: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for browser tasks. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

File Operations - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats file operations inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about file operations, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what file operations must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for file operations: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for file operations. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Calendar Workflow - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats calendar workflow inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about calendar workflow, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what calendar workflow must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for calendar workflow: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for calendar workflow. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Email Workflow - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats email workflow inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about email workflow, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what email workflow must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for email workflow: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for email workflow. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Research Agent - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats research agent inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about research agent, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what research agent must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for research agent: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for research agent. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Retrieval Agent - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats retrieval agent inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about retrieval agent, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what retrieval agent must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for retrieval agent: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for retrieval agent. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Critique Agent - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats critique agent inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about critique agent, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what critique agent must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for critique agent: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for critique agent. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Coding Agent - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats coding agent inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about coding agent, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what coding agent must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for coding agent: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for coding agent. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Design Agent - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats design agent inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about design agent, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what design agent must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for design agent: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for design agent. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Personal Assistant - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats personal assistant inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about personal assistant, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what personal assistant must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for personal assistant: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for personal assistant. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Business Assistant - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats business assistant inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about business assistant, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what business assistant must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for business assistant: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

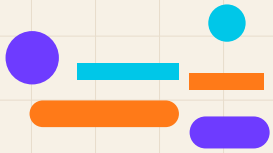
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for business assistant. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Guarded Autonomy - Interface Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats guarded autonomy inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about guarded autonomy, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what guarded autonomy must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for guarded autonomy: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

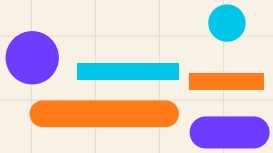
MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for guarded autonomy. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Human-In-The-Loop - Evaluation Engine



ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats human-in-the-loop inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about human-in-the-loop, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what human-in-the-loop must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for human-in-the-loop: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for human-in-the-loop. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Condition Checks - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats condition checks inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about condition checks, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what condition checks must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for condition checks: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for condition checks. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Automation Limits - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats automation limits inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about automation limits, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what automation limits must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for automation limits: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for automation limits. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Status Report - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats status report inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about status report, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what status report must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for status report: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for status report. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Failure Recovery - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats failure recovery inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about failure recovery, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what failure recovery must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for failure recovery: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for failure recovery. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Runbook - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats runbook inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about runbook, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what runbook must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for runbook: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for runbook. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Memory Update - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats memory update inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about memory update, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what memory update must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for memory update: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for memory update. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Multi-Agent Debate - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats multi-agent debate inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about multi-agent debate, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what multi-agent debate must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for multi-agent debate: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for multi-agent debate. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Task Scheduling - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats task scheduling inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about task scheduling, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what task scheduling must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for task scheduling: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for task scheduling. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Agent Qa - Evaluation Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats agent QA inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about agent QA, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what agent QA must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for agent QA: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for agent QA. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Agentic Court - Agentic Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats agentic court inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about agentic court, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what agentic court must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for agentic court: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for agentic court. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version

Final Os Kernel - Interface Engine

ChatGPT MCD OS chunk Court-tested

OPERATING THESIS

This chunk treats final OS kernel inside Agentic Execution & Automation as an operable ChatGPT unit, not a decorative feature. The user begins with an intent, a context boundary, and an output contract; ChatGPT is then used to transform ambiguity into a draft, plan, explanation, code path, analysis, critique, or decision support. The operating rule is simple: every answer must expose its assumptions, preserve user agency, and end with a usable next action or a repairable artifact.

MECHANISM GRAPH

Mechanism graph. Input: a user goal about final OS kernel, plus constraints, audience, source status, privacy level, and success metric. Transformation: ChatGPT parses the task, asks or infers missing structure, uses available tools when appropriate, composes a candidate output, and marks uncertainty. Mediators: prompt clarity, context quality, model capability, tool access, source freshness, and file quality. Output: a usable artifact plus a traceable quality test. Failure condition: the response sounds fluent but cannot be checked, reused, taught, or safely acted on.

DESIGN PROTOCOL

1) Name the job: define what final OS kernel must accomplish. 2) Define the audience and format before asking for content. 3) Add source status: official, provided, inferred, or needs verification. 4) Ask for a first model, not a final answer. 5) Run a critique pass against hallucination, reductionism, and weak operation. 6) Request a repaired version with explicit assumptions. 7) Convert the result into a reusable template, checklist, or prompt cell. 8) Store the improved pattern in a project, library, or version log.

DESTRUCTION COURT

Destruction court for final OS kernel: reject outputs that hide plan limits, invent product behavior, treat a model as an authority, collapse uncertainty into confidence, over-disclose sensitive data, or produce generic advice that cannot be evaluated. Repair by adding a source ladder, a user-agency warning, a minimal test, and a narrower claim. The best answer is not the longest answer; it is the answer whose operational effect can be inspected.

MCD / QMCD NOTE

MCD/QMCD note. D = distinctions between prompt, context, tool, source, and output. I = integration with Agentic Execution & Automation without mixing product fact and theory. T = sensitivity to version changes and plan variance. A = user retains control of decisions. P = the workflow can predict better outputs under clearer constraints. C = the chunk can regenerate prompts, rubrics, and examples. N = hype, vague intelligence language, and unsupported capability claims.

AGENTIC PROMPT CELL

Prompt cell: Use ChatGPT as a Agentic Execution & Automation operating assistant for final OS kernel. First restate my objective, constraints, privacy risk, and success metric. Then produce a structured output. Then attack your own output through hallucination, reductionism, false novelty, and weak operationalization courts. Finally repair it into a reusable checklist and give me a compact prompt I can reuse in a project.

QUALITY LOOP Brief -> Prompt -> Context -> Answer -> Court -> Repair -> Memory -> Version