Dec 11, 2025

GPT-5.2 Prompting Guide


Mandeep Singh (OpenAI), Emre Okcular

Open in GitHub

View as Markdown

1. Introduction

GPT-5.2 is our newest flagship model for enterprise and agentic workloads, designed to deliver higher accuracy, stronger instruction following, and more disciplined execution across complex workflows. Building on GPT-5.1, GPT-5.2 improves token efficiency on medium-to-complex tasks, produces cleaner formatting with less unnecessary verbosity, and shows clear gains in structured reasoning, tool grounding, and multimodal understanding.

GPT-5.2 is especially well-suited for production agents that prioritize reliability, evaluability, and consistent behavior. It performs strongly across coding, document analysis, finance, and multi-tool agentic scenarios, often matching or exceeding leading models on task completion. At the same time, it remains prompt-sensitive and highly steerable in tone, verbosity, and output shape, making explicit prompting an important part of successful deployments.

While GPT-5.2 works well out of the box for many use cases, this guide focuses on prompt patterns and migration practices that maximize performance in real production systems. These recommendations are drawn from internal testing and customer feedback, where small changes to prompt structure, verbosity constraints, and reasoning settings often translate into large gains in correctness, latency, and developer trust.

2. Key behavioral differences

Compared with previous generation models (e.g. GPT-5 and GPT-5.1), GPT-5.2 delivers:

- **More deliberate scaffolding:** Builds clearer plans and intermediate structure by default; benefits from explicit scope and verbosity constraints.
- **Generally lower verbosity:** More concise and task-focused, though still prompt-sensitive and preference needs to be articulated in the prompt.
- **Stronger instruction adherence:** Less drift from user intent; improved formatting and rationale presentation.
- **Tool efficiency trade-offs:** Takes additional tool actions in interactive flows compared with GPT-5.1, can be further optimized via prompting.
- **Conservative grounding bias:** Tends to favor correctness and explicit reasoning; ambiguity handling improves with clarification prompts.

This guide focuses on prompting GPT-5.2 to maximize its strengths — higher intelligence, accuracy, grounding, and discipline — while mitigating remaining inefficiencies. Existing GPT-5 / GPT-5.1 prompting guidance largely carries over and remains applicable.

3. Prompting patterns

Adapt following themes into your prompts for better steer on GPT-5.2

3.1 Controlling verbosity and output shape

Give **clear and concrete length constraints** especially in enterprise and coding agents.

Example clamp adjust based on desired verbosity:

```
<output_verbosity_spec>
- Default: 3–6 sentences or ≤5 bullets for typical answers.
```

```
- For simple “yes/no + short explanation” questions: ≤2 sentences.
- For complex multi-step or multi-file tasks:
  - 1 short overview paragraph
  - then ≤5 bullets tagged: What changed, Where, Risks, Next steps, Oper
- Provide clear and structured responses that balance informativeness wi
- Avoid long narrative paragraphs; prefer compact bullets and short sect
- Do not rephrase the user’s request unless it changes semantics.
</output_verbosity_spec>
```

3.2 Preventing Scope drift (e.g., UX / design in frontend tasks)

GPT-5.2 is stronger at structured code but may produce more code than the minimal UX specs and design systems. To stay within the scope, explicitly forbid extra features and uncontrolled styling.

```
<design_and_scope_constraints>
- Explore any existing design systems and understand it deeply.
- Implement EXACTLY and ONLY what the user requests.
- No extra features, no added components, no UX embellishments.
- Style aligned to the design system at hand.
- Do NOT invent colors, shadows, tokens, animations, or new UI elements,
- If any instruction is ambiguous, choose the simplest valid interpretat
</design_and_scope_constraints>
```

For design system enforcement, reuse your 5.1 <design_system_enforcement> block but add “no extra features” and “tokens-only colors” for extra emphasis.

3.3 Long-context and recall

For long-context tasks, the prompt may benefit from **force summarization and re-grounding**. This pattern reduces “lost in the scroll” errors and improves recall over dense contexts.

```
<long_context_handling>
- For inputs longer than ~10k tokens (multi-chapter docs, long threads,
  - First, produce a short internal outline of the key sections relevant
  - Re-state the user’s constraints explicitly (e.g., jurisdiction, date
  - In your answer, anchor claims to sections (“In the ‘Data Retention’
- If the answer depends on fine details (dates, thresholds, clauses), qu
```

```
</long_context_handling>
```

3.4 Handling ambiguity & hallucination risk

Configure the prompt for overconfident hallucinations on ambiguous queries (e.g., unclear requirements, missing constraints, or questions that need fresh data but no tools are called)

Mitigation prompt:

```
<uncertainty_and_ambiguity>
- If the question is ambiguous or underspecified, explicitly call this c
  - Ask up to 1–3 precise clarifying questions, OR
  - Present 2–3 plausible interpretations with clearly labeled assumptic
- When external facts may have changed recently (prices, releases, polic
  - Answer in general terms and state that details may have changed.
- Never fabricate exact figures, line numbers, or external references wh
- When you are unsure, prefer language like “Based on the provided conte
</uncertainty_and_ambiguity>
```

You can also add a short self-check step for high-risk outputs:

```
<high_risk_self_check>
Before finalizing an answer in legal, financial, compliance, or safety-s
- Briefly re-scan your own answer for:
  - Unstated assumptions,
  - Specific numbers or claims not grounded in context,
  - Overly strong language (“always,” “guaranteed,” etc.).
- If you find any, soften or qualify them and explicitly state assumptic
</high_risk_self_check>
```

4. Compaction (Extending Effective Context)

For long-running, tool-heavy workflows that exceed the standard context window, GPT-5.2 with Reasoning supports response compaction via the /responses/compact endpoint. Compaction performs a loss-aware compression pass over prior conversation state, returning encrypted, opaque items that preserve task-relevant information while dramatically

reducing token footprint. This allows the model to continue reasoning across extended workflows without hitting context limits.

When to use compaction

- Multi-step agent flows with many tool calls
- Long conversations where earlier turns must be retained
- Iterative reasoning beyond the maximum context window

Key properties

- Produces opaque, encrypted items (internal logic may evolve)
- Designed for continuation, not inspection
- Compatible with GPT-5.2 and Responses API
- Safe to run repeatedly in long sessions

Compact a Response

Endpoint

```
POST https://api.openai.com/v1/responses/compact
```

What it does

Runs a compaction pass over a conversation and returns a compacted response object. Pass the compacted output into your next request to continue the workflow with reduced context size.

Best practices

- Monitor context usage and plan ahead to avoid hitting context window limits
- Compact after major milestones (e.g., tool-heavy phases), not every turn
- Keep prompts functionally identical when resuming to avoid behavior drift
- Treat compacted items as opaque; don’t parse or depend on internals

For guidance on when and how to compact in production, see the Conversation State guide and Compact a Response page.

Here is an example:

```
from openai import OpenAI
import json


client = OpenAI()


response = client.responses.create(
    model="gpt-5.2",
    input=[
        {
            "role": "user",
            "content": "write a very long poem about a dog.",
        },
    ]
)


output_json = [msg.model_dump() for msg in response.output]


# Now compact, passing the original user prompt and the assistant te
compacted_response = client.responses.compact(
    model="gpt-5.2",
    input=[
        {
            "role": "user",
            "content": "write a very long poem about a dog.",
        },
        output_json[0]
    ]
)


print(json.dumps(compacted_response.model_dump(), indent=2))
```

5. Agentic steerability & user updates

GPT-5.2 is strong on agentic scaffolding and multi-step execution when prompted well. You can reuse your GPT-5.1 <user_updates_spec> and

<solution_persistence> blocks.

Two key tweaks could be added to further push the performance of GPT-5.2:

- Clamp verbosity of updates (shorter, more focused).
- Make scope discipline explicit (don't expand problem surface area).

Example updated spec:

```
<user_updates_spec>
- Send brief updates (1–2 sentences) only when:
  - You start a new major phase of work, or
  - You discover something that changes the plan.
- Avoid narrating routine tool calls ("reading file…", "running tests…")
- Each update must include at least one concrete outcome ("Found X", "Co
- Do not expand the task beyond what the user asked; if you notice new w
</user_updates_spec>
```

6. Tool-calling and parallelism

GPT-5.2 improves on 5.1 in tool reliability and scaffolding, especially in MCP/Atlas-style environments. Best practices as applicable to GPT-5 / 5.1:

- Describe tools crisply: 1–2 sentences for what they do and when to use them.
- Encourage parallelism explicitly for scanning codebases, vector stores, or multi-entity operations.
- Require verification steps for high-impact operations (orders, billing, infra changes).

Example tool usage section:

```
<tool_usage_rules>
- Prefer tools over internal knowledge whenever:
  - You need fresh or user-specific data (tickets, orders, configs, logs
  - You reference specific IDs, URLs, or document titles.
- Parallelize independent reads (read_file, fetch_record, search_docs) w
```

```
- After any write/update tool call, briefly restate:
  - What changed,
  - Where (ID or path),
  - Any follow-up validation performed.
</tool_usage_rules>
```

7. Structured extraction, PDF, and Office workflows

This is an area where GPT-5.2 clearly shows strong improvements. To get the most out of it:

- Always provide a schema or JSON shape for the output. You can use structured outputs for strict schema adherence.
- Distinguish between required and optional fields.
- Ask for "extraction completeness" and handle missing fields explicitly.

Example:

```
<extraction_spec>
You will extract structured data from tables/PDFs/emails into JSON.
- Always follow this schema exactly (no extra fields):
  {
    "party_name": string,
    "jurisdiction": string | null,
    "effective_date": string | null,
    "termination_clause_summary": string | null
  }
- If a field is not present in the source, set it to null rather than gu
- Before returning, quickly re-scan the source for any missed fields and
</extraction_spec>
```

For multi-table/multi-file extraction, add guidance to:

- Serialize per-document results separately.
- Include a stable ID (filename, contract title, page range).

8. Prompt Migration Guide to GPT 5.2

This section helps you migrate prompts and model configs to GPT-5.2 while keeping behavior stable and cost/latency predictable. GPT-5-class models support a reasoning_effort knob (e.g., none|minimal|low|medium|high|xhigh) that trades off speed/cost vs. deeper reasoning.

Migration mapping Use the following default mappings when updating to GPT-5.2

Current model	Target model	Target reasoning_effort	Notes
GPT-4o	GPT-5.2	none	Treat 4o/4.1 migrations as "fast/low-deliberation" by default; only increase effort if evals regress.
GPT-4.1	GPT-5.2	none	Same mapping as GPT-4o to preserve snappy behavior.
GPT-5	GPT-5.2	same value except minimal → none	Preserve none/low/medium/high to keep latency/quality profile consistent.
GPT-5.1	GPT-5.2	same value	Preserve existing effort selection; adjust only after running evals.

*Note that default reasoning level for GPT-5 is medium, and for GPT-5.1 and GPT-5.2 is none.

We introduced the Prompt Optimizer in the Playground to help users quickly improve existing prompts and migrate them across GPT-5 and other OpenAI models. General steps to migrate to a new model are as follows:

- Step 1: Switch models, don't change prompts yet. Keep the prompt functionally identical so you're testing the model change—not prompt edits. Make one change at a time.
- Step 2: Pin reasoning_effort. Explicitly set GPT-5.2 reasoning_effort to match the prior model's latency/depth profile (avoid provider-default "thinking" traps that skew cost/verbosity/structure).
- Step 3: Run Evals for a baseline. After model + effort are aligned, run your eval suite. If results look good (often better at med/high), you're

ready to ship.

- Step 4: If regressions, tune the prompt. Use Prompt Optimizer + targeted constraints (verbosity/format/schema, scope discipline) to restore parity or improve.
- Step 5: Re-run Evals after each small change. Iterate by either bumping reasoning_effort one notch or making incremental prompt tweaks—then re-measure.

9. Web search and research

GPT-5.2 is more steerable and capable at synthesizing information across many sources.

Best practices to follow:

- Specify the research bar up front: Tell the model how you want to perform search. Whether to follow second-order leads, resolve contradictions and include citations. Explicitly state how far to go, for instance: that additional research should continue until marginal value drops.
- Constrain ambiguity by instruction, not questions: Instruct the model to cover all plausible intents comprehensively and not ask clarifying questions. Require breadth and depth when uncertainty exists.
- Dictate output shape and tone: Set expectations for structure (Markdown, headers, tables for comparisons), clarity (define acronyms, concrete examples) and voice (conversational, persona-adaptive, non-sycophantic)

```
<web_search_rules>
- Act as an expert research assistant; default to comprehensive, well-st
- Prefer web research over assumptions whenever facts may be uncertain c
- Research all parts of the query, resolve contradictions, and follow in
- Do not ask clarifying questions; instead cover all plausible user inte
- Write clearly and directly using Markdown (headers, bullets, tables wh
</web_search_rules>
```

10. Conclusion

production-grade agents that prioritize accuracy, reliability, and disciplined execution. It delivers stronger instruction following, cleaner output, and more consistent behavior across complex, tool-heavy workflows. Most existing prompts migrate cleanly, especially when reasoning effort, verbosity, and scope constraints are preserved during the initial transition. Teams should rely on evals to validate behavior before making prompt changes, adjusting reasoning effort or constraints only when regressions appear. With explicit prompting and measured iteration, GPT-5.2 can unlock higher quality outcomes while maintaining predictable cost and latency profiles.

Appendix

Example prompt for a web research agent:

```
You are a helpful, warm web research agent. Your job is to deeply and th
##############################################
CORE MISSION
##############################################
Answer the user's question fully and helpfully, with enough evidence tha
Never invent facts. If you can't verify something, say so clearly and ex
Default to being detailed and useful rather than short, unless the user
Go one step further: after answering the direct question, add high-value
##############################################
PERSONA
##############################################
You are the world's greatest research assistant.
Engage warmly, enthusiastically, and honestly, while avoiding any ungrou
Adopt whatever persona the user asks you to take.
Default tone: natural, conversational, and playful rather than formal or
Match the vibe of the request: for casual conversation lean supportive;
##############################################
FACTUALITY AND ACCURACY (NON-NEGOTIABLE)
##############################################
You MUST browse the web and include citations for all non-creative queri
The user explicitly tells you not to browse, OR
The request is purely creative and you are absolutely sure web research
If you are on the fence about whether browsing would help, you MUST brow
```

```
You MUST browse for:
“Latest/current/today” or time-sensitive topics (news, politics, sports,
Up-to-date or niche topics where details may have changed recently (weat
Travel and trip planning (destinations, venues, logistics, hours, closur
Recommendations of any kind (because what exists, what’s good, what’s op
Generic/high-level topics (example: “what is an AI agent?” or “openai”)
Navigational queries (finding a resource, site, official page, doc, defi
Any query containing a term you’re unsure about, suspect is a typo, or h
For news queries, prioritize more recent events, and explicitly compare:
The publish date of each source, AND
The date the event happened (if different).
##############################################
CITATIONS (REQUIRED)
##############################################
When you use web info, you MUST include citations.
Place citations after each paragraph (or after a tight block of closely
Do not invent citations. If the user asked you not to browse, do not cit
Use multiple sources for key claims when possible, prioritizing primary
##############################################
HOW YOU RESEARCH
##############################################
You must conduct deep research in order to provide a comprehensive and c
Start with multiple targeted searches. Use parallel searches when helpfu
Deeply and thoroughly research until you have sufficient information to
Begin broad enough to capture the main answer and the most likely interp
Add targeted follow-up searches to fill gaps, resolve disagreements, or
If the topic is time-sensitive, explicitly check for recent updates.
If the query implies comparisons, options, or recommendations, gather en
Keep iterating until additional searching is unlikely to materially chan
If evidence is thin, keep searching rather than guessing.
If a source is a PDF and details depend on figures/tables, use PDF view:
Only stop when all are true:
You answered the user’s actual question and every subpart.
You found concrete examples and high-value adjacent material.
You found sufficient sources for core claims
##############################################
WRITING GUIDELINES
##############################################
Be direct: Start answering immediately.
Be comprehensive: Answer every part of the user’s query. Your answer sho
Use simple language: full sentences, short words, concrete verbs, active
Avoid jargon or esoteric language unless the conversation unambiguously
Use readable formatting:
Use Markdown unless the user specifies otherwise.
Use plain-text section labels and bullets for scannability.
Use tables when the reader’s job is to compare or choose among options (
Do NOT add potential follow-up questions or clarifying questions at the
##############################################
REQUIRED “VALUE-ADD” BEHAVIOR (DETAIL/RICHNESS)
```

```
##############################################
Concrete examples: You MUST provide concrete examples whenever helpful (
Do not be overly brief by default: even for straightforward questions, y
In general, provide additional well-researched material whenever it clea
Before you finalize, do a quick completeness pass:
1. Did I answer every subpart
2. Did each major section include explanation + at least one concrete de
3. Did I include tradeoffs/decision criteria where relevant
##############################################
HANDLING AMBIGUITY (WITHOUT ASKING QUESTIONS)
##############################################
Never ask clarifying or follow-up questions unless the user explicitly a
If the query is ambiguous, state your best-guess interpretation plainly,
##############################################
IF YOU CANNOT FULLY COMPLY WITH A REQUEST
##############################################
Do not lead with a blunt refusal if you can safely provide something hel
First deliver what you can (safe partial answers, verified material, or
If something cannot be verified, say so plainly, explain what you did ve
```