ChatGPT


```
{
  "citations": [
    "Augment Code Blog – *How to build your agent: 11 prompting techniques for better AI agents* (May 21 2025) [1]",
    "Cursor Documentation – *Rules* (2023) [2]",
    "Cursor Documentation – *Working with Context* (2023) [3]",
    "Cursor Documentation – *Codebase Indexing* (2023) [4]",
    "Cursor Documentation – *Keyboard Shortcuts* (2023) [5]",
    "Cursor Community Forum – *Best Practices: .cursorrules* (Jan 15 2025) [6]",
    "Cursor Community Forum – *Applying changes to a large file fails* (Feb 19 2025) [7]",
    "Cursor Community Forum – *Cursor timeout agent mode* (Jan 27 2025) [8]",
    "Stephen A. Fuqua – *Grudgingly Accepting AI Coding Assistants* (Jan 14 2025) [9]",
    "HiddenLayer Research – *Hidden Prompt Injections Hijacking Cursor* (Jul 31 2025) [10]",
    "DEV Community (ImagineX) – *Lead Agents with Prompts* (Apr 28 2024) [11]",
    "DEV Community (M. Tamrkar) – *Mastering Cursor AI (2025 Edition)* (Apr 22 2025) [12]",
    "Reddit (r/ChatGPTCoding) – *Prompts for high-quality code generation* (2024) [13]"
  ],
  "agent_prompting": {
    "patterns": [
      "Provide complete and relevant context in prompts to improve the model's accuracy [14] (Augment Code, May 21 2025).",

"Set the stage for the AI by stating its role and available tools (e.g. that it can read/write code) [15] (Augment Code, May 21 2025).",

"Keep instructions consistent across system prompts, tool descriptions, and user messages to avoid confusing the agent [16] (Augment Code, May 21 2025).",

"Break complex tasks into a planning step and an execution step instead of one big prompt [11] (DEV Community, Apr 28 2024).",
      "Don't fear longer prompts – provide thorough details. Models can handle long context and benefit from clarity [17] (Augment Code, May 21 2025).",
      "Emphasize critical instructions at the beginning or end of the prompt where the model gives them more weight [18] (Augment Code, May 21 2025).",
      "Recognize prompting plateaus: if tweaking wording no longer improves results, consider altering strategy or providing more/different context [19] (Augment Code, May 21 2025)."
    ],
    "templates": [
      "Use multi-step prompt templates (e.g. "Plan then Apply") to let the
```

```
agent first outline a solution, then generate code in minimal diffs [20] (DEV Community, Apr 28 2024).",
      "Save reusable prompt files for common workflows (like test generation or component scaffolding) to standardize how you instruct the agent [21] (DEV Community, Apr 28 2024).",
      "Include checklists or bullet points in your prompt (requirements, constraints, etc.) so the agent addresses each aspect methodically [13] (Reddit, 2024).",
      "Leverage Cursor's persistent rules (Project Rules or AGENTS.md) to provide default guidelines and examples that shape the agent's behavior across sessions [22] [23] (Cursor Docs, 2023).",
      "For test-driven development, prompt the agent to draft unit tests (covering normal and edge cases) before writing implementation code [24] (Reddit, 2024)."
    ],
    "anti_patterns": [
      "Avoid vague prompts with generic requests – be specific about desired changes or outcomes [25] (Cursor Docs, 2023).",
      "Don't include dynamic or changing data (like timestamps or volatile state) in the system prompt; it can confuse the model if the context drifts [26] (Augment Code, May 21 2025).",
      "Resist giving the agent free rein to run tools/commands globally without checks; always require confirmation for potentially destructive actions [27] (DEV Community, Apr 28 2024).",
      "Do not assume the AI will infer context you haven't provided – insufficient context leads to hallucinations or irrelevant outputs [28] (Cursor Docs, 2023).",
      "Never put sensitive information (API keys, passwords) into the prompt or context – treat all AI-visible content as public [10] (HiddenLayer, Jul 31 2025).",

"Avoid tackling an entire large task in one go. It's an anti-pattern to ask for a massive multi-file refactor or feature in one prompt; break it into smaller subtasks to maintain control and clarity [29] (Tamrkar, Apr 22 2025)."
    ]
  },
  "workflows": {
    "multi_file_edits": [
      "Review and apply multi-file changes one file at a time. Use the diff view to inspect each file's modifications and open each file before applying changes to ensure they go to the correct place [30] (Cursor Forum, Feb 2025).",
      "Use version control branches for multi-file edits. Commit incremental changes frequently so you can revert if the AI's suggestions introduce issues [12] (Tamrkar, Apr 22 2025).",
      "Add all relevant files to the context for cross-file changes. For example, use @file or @folder to include dependent modules so the agent is aware of references across files [31] [32] (Cursor Docs, 2023).",
      "Limit the scope of each edit session to a single feature or fix. Cursor works best with small, incremental changes – large monolithic edits are more error-prone [29] (Tamrkar, Apr 22 2025).",
      "Always review the combined diff that the agent proposes for multiple
```

```
files, and double-check that each change is necessary and correct before accepting [12] (Tamrkar, Apr 22 2025)."
    ],
    "refactor_flows": [
      "When refactoring, preserve the external API and focus on internal improvements. Instruct the agent not to change public function signatures or behavior [33] (Tamrkar, Apr 2025).",
      "Refactor in small steps. Tackle one module or component at a time and test after each change, rather than refactoring the whole codebase in one sweep [34] (Tamrkar, Apr 2025).",

"After refactoring, run all relevant tests (or have the agent run them) to ensure nothing broke. Cursor can assist by generating or updating tests for the refactored code [35] (Tamrkar, Apr 2025).",

"Use Cursor's rules to enforce architecture or style constraints during refactoring (e.g., a rule to always use a certain pattern) so the AI adheres to your design principles [23] [22] (Cursor Docs, 2023).",
      "Leverage version control: commit each refactor step. If the AI makes an incorrect change, having granular commits makes it easier to roll back that specific change [36] (DEV Community, 2024)."
    ],
    "tdd_flows": [
      "Start with a failing test. Have the agent write unit tests that demonstrate the feature or bug to fix (these should initially fail) [24] (Reddit, 2024).",
      "Review and possibly refine the generated tests to ensure they cover edge cases and express the intended behavior clearly [24] (Reddit, 2024).",
      "Next, ask the agent to implement the code to make those tests pass. This encourages a focus on just enough code to satisfy the tested requirements [37] (Fuqua, Jan 2025).",
      "Run the test suite using Cursor's integrated terminal or your own environment. Provide the agent with the test results (or let it see the failures) so it can iterate on the solution [38] (Cursor Docs, 2023).",
      "Keep the red/green/refactor cycle: generate tests (red), implement code to pass (green), then consider refactoring with the safety of tests. Cursor's agent can assist at each step of this cycle [37] [35] (Fuqua, Jan 2025; Tamrkar, Apr 2025)."
    ],
    "bugfix_flows": [
      "Reproduce the bug with a test or detailed log. Provide the error message, stack trace, or a failing scenario to the agent as context [39] (Cursor Docs, 2023).",

"Ask the agent to diagnose the cause of the failure. Often, Cursor will search the codebase or analyze the provided context to hypothesize the root cause [40] (Cursor Docs, 2023).",
      "Once the cause is identified, have the agent suggest a fix. It can edit the relevant files or propose a patch; review this diff carefully to ensure it addresses the bug and doesn't introduce side effects [12] (Tamrkar, Apr 2025).",
```

```
      "After applying the fix, run the relevant tests or reproduction steps
again (Cursor can execute them via its terminal tool) to confirm the bug is
resolved [38] (Cursor Docs, 2023).",
      "If the issue persists or new errors arise, iterate: provide the new
output or test failures to the agent and refine the fix. This loop continues
until tests pass and the bug is fully fixed [38] (Cursor Docs, 2023)."
    ],
    "keyboard_cheatsheet": [
      "`Ctrl+Shift+L` – Add selected text as context to chat (includes it in
the prompt for the AI) [41] (Cursor Docs).",

"`Ctrl+Shift+K` – Open an Inline Edit with the selected code pre-loaded
(allows asking for changes in-place) [42] (Cursor Docs).",
      "`Ctrl+Shift+Backspace` – Cancel the ongoing AI generation (stop the
agent's response in both chat or inline edit modes) [43] (Cursor Docs).",
      "`Ctrl+Enter` (when AI suggestions are present) – Accept all suggested
code changes; conversely, `Ctrl+Backspace` rejects the suggestions [44] (Cursor
Docs).",
      "`Ctrl+Shift+P` – Open the Command Palette (access Cursor commands and
VS Code commands through search) [45] (Cursor Docs).",
      "After copying code: `Ctrl+V` in chat inserts it as a reference
(context link), whereas `Ctrl+Shift+V` pastes the actual code into the chat
input [46] (Cursor Docs)."
    ]
  },
  "context": {
    "scoping_strategies": [
      "Use the `@`-symbol to precisely pull in context: `@code` for a
specific function or symbol, `@file` for an entire file, `@folder` for a
directory [31] (Cursor Docs, 2023).",
      "Exclude irrelevant or large files from indexing with `.cursorignore`/
`.gitignore` so they don't distract the AI [47] (Cursor Docs, 2023).",

"Manually provide context you know is needed (key files, configs, logs)
rather than relying solely on automatic context gathering [48] (Cursor Docs,
2023).",
      "Keep prompts focused on one intent at a time. If a task involves
distinct parts, handle them in separate conversations or steps to reduce
context dilution [29] (Tamrkar, Apr 2025).",
      "In multi-repo or monorepo setups, take advantage of multi-root
workspace support – Cursor will index each project folder. Ensure project-
specific rules are placed in each root if needed [49] (Cursor Docs, 2023)."
    ],
    "repo_navigation": [
      "Use Cursor's indexed search to ask high-level questions about the
codebase (e.g., "Where is the cache initialized?") – the agent can retrieve
relevant file snippets thanks to semantic indexing [50] (Cursor Docs, 2023).",
      "Employ PR/commit search by referencing them: `@[PR number]` or
`@[commit hash]` will fetch those changes into context, including commit
messages and code diffs [4] (Cursor Docs, 2023).",
      "Keep design docs and READMEs in your repository. The agent can read
```

```
them (via context or tools) to better understand the project's architecture and goals [51] (Cursor Forum, Jan 2025).",
      "Use `@symbolName` (function or class name) to have Cursor quickly navigate to that definition and include it in the conversation [31] (Cursor Docs, 2023).",
      "Check which files are indexed via *Indexing & Docs* settings (`View included files`) if you suspect Cursor isn't aware of some code. Add any missing important files to the index or open them so they get loaded [52] (Cursor Docs, 2023)."
    ],
    "long_file_handling": [

"Be aware of Cursor's 250-line read limit for unmapped files. If a file is long, attach it or ask the agent to read it in chunks (e.g., lines 1–250, 251–500, etc.) [53] [54] (Cursor Forum, Feb 2025).",
      "For extremely large files (tens of thousands of lines), consider splitting the task: extract the relevant section to a separate file to edit, or use search-and-replace or CLI tools via Cursor to make targeted changes [55] [7] (Cursor Forum, Feb 2025).",
      "If applying changes to a huge file fails or the agent gets stuck, try using the diff mode. Have Cursor generate a patch/diff for that file and apply the changes manually in chunks [30] [7] (Cursor Forum, Feb 2025).",
      "Use models with larger context windows for big files. If you have access to a model like Claude or Gemini with more tokens, switch to it for tasks involving very large files to reduce chances of context overflow [56] (Cursor Forum, Feb 2025).",
      "Watch out for editor limitations with very high line numbers. If the inline edit UI behaves oddly past ~50k lines (e.g., not opening at the correct spot), you may need to resort to external scripting or break the edit into smaller pieces [57] [58] (Cursor Forum, Feb 2025)."
    ]
  },
  "guardrails": {
    "quality_checks": [
      "Always review the AI-generated diff line-by-line before merging the changes [12] (Tamrkar, Apr 2025).",
      "Run linters and static analyzers on AI-written code to catch syntax errors, style issues, or obvious bugs [9] (Fuqua, Jan 2025).",
      "Execute the test suite after the AI's changes. Ensure all tests pass, and add new tests for any new functionality the AI introduced [35] (Tamrkar, Apr 2025).",
      "Apply security scanning (e.g., CodeQL or similar tools) on the AI's code contributions to detect vulnerabilities or insecure code patterns [59] (Fuqua, Jan 2025).",

"Use peer review for AI-generated code. Treat the AI's output like code from a junior developer – require another developer to approve it via code review before it goes into production [60] (Fuqua, Jan 2025)."
    ],
    "security_safety": [
```

```
"Never paste API keys, passwords, or sensitive data into Cursor. The AI might include them in output or they could be logged [10] (HiddenLayer, Jul 2025).",

"Use Cursor's denylist for commands to prevent the agent from executing dangerous operations. For example, add destructive commands like `rm -rf` or network calls like `curl` to the denied tools, so the agent must ask for permission [61] (HiddenLayer, Jul 2025).",
      "Remain vigilant for prompt injection in your codebase. Check README files or other text that Cursor might read for any hidden instructions (e.g., invisible Unicode) that could hijack the agent [10] [62] (HiddenLayer, Jul 2025).",
      "Prefer to run build or deploy commands manually rather than letting the agent do it blindly. You can have the agent suggest the command, then you execute it and feed back the output for analysis, which avoids the agent mismanaging your runtime environment [63] (Cursor Forum, Apr 2025).",
      "Keep Cursor up-to-date. Security patches (like the fix for the prompt injection vulnerabilities in Cursor 1.3) are released in updates [64] (HiddenLayer, Jul 2025). Regularly install the latest version to benefit from these fixes."
    ],
    "review_checklists": [
      "Verify that the change meets all requirements: does it solve the stated problem or implement the feature as described (no missing functionality)? [65] (Reddit, 2024).",

"Check code quality: Is the code following project style guidelines and best practices (naming, formatting, patterns)? [66] (Reddit, 2024).",
      "Think about edge cases and error conditions: Does the code handle invalid inputs or potential failures gracefully (no uncaught exceptions)? [67] [65] (Reddit, 2024).",
      "Consider performance: Ensure the AI didn't introduce an obvious inefficiency (e.g., O(n^2) loop where unnecessary) or memory-heavy operation that could be optimized [68] (Reddit, 2024).",
      "Examine security implications: No sensitive info is exposed, and the code doesn't introduce injection vulnerabilities or unsafe use of external data (especially for web apps) [65] (Reddit, 2024).",
      "Ensure maintainability: The code should be reasonably understandable with clear logic and comments where needed, so future developers (or AI) can work with it easily [69] [65] (Reddit, 2024)."
    ]
  },
  "troubleshooting": {
    "failure_modes": [
      "**Agent ignoring rules**: If the agent isn't following your .cursorrules/project rules (especially in Compose mode), ensure the rules file is in the project root and note that rules apply mainly to chat/agent modes, not the freeform editor [70] (Cursor Forum, Jan 2025).",
      "**Stuck 'generating' or timeouts**: The agent may hang or time out on very large tasks or after running for a long time [71]. This often indicates it hit a limit (time or tokens) and couldn't complete.",
      "**Repeated apologies/loops**: Sometimes the agent gets into a loop of
```

```
trying and failing (e.g., on very large files) [72]. It might repeatedly
apologize or retry without progress.",
      "**Wrong file edits**: In multi-file operations, the agent might apply
changes to the wrong file if those files weren't open or properly referenced,
leading to "canceled" edits or changes in the wrong place [30] (Cursor Forum,
Feb 2025).",
      "**Terminal tool issues**: The agent might not capture output from
long-running commands, or commands might be interrupted (the agent may send a
Ctrl+C prematurely) [73] [74] (Cursor Forum, Apr 2025). This results in
incomplete execution and the agent stuck waiting.",
      "**Large context overload**: If you add too many large files to
context, the model might start hallucinating or ignoring some context due to
token limit. It may produce irrelevant answers or partial code in such
cases."
    ],
    "recovery_steps": [
      "If the agent times out or stalls, try breaking the task into smaller
parts. Refresh the conversation (start a new chat) and tackle sub-tasks one
by one rather than a giant request [8] (Cursor Staff, Feb 2025).",
      "For an agent caught in a failure loop, use the
"Cancel" (`Ctrl+Shift+Backspace`) to stop it [43], then rephrase your last
instruction or simplify the context (remove less relevant parts) before
trying again.",
      "If .cursorrules aren't being applied, you can manually remind the
agent of key rules in the prompt. Also ensure you're on the latest Cursor
version, as rule application bugs get fixed quickly (and consider migrating
legacy .cursorrules to the new project rules format) [75] [64].",
      "When changes apply to the wrong file or not at all, open the target
file and re-run the change in that file's context. In multi-file diffs, apply
them one file at a time as a workaround [30] (Cursor Forum, Feb 2025).",

"If the agent introduced an error or regression, use `git diff` or your VCS
to pinpoint the problematic change, then describe that diff to Cursor and ask
for a fix specifically for that regression. This scopes the AI on the narrow
problem.",
      "For terminal tool problems (commands not finishing), run the command
yourself in a separate terminal. Then take the output (or note any error) and
feed it back into Cursor so the agent can continue based on the actual result
 [63] (Cursor Staff, Apr 2025)."
    ],
    "performance_tips": [
      "Use a faster model for drafts and a smarter model for finalization.
For example, do initial coding with a 3.5-tier model for speed, then switch
to GPT-4 for thoroughness when reviewing or refining (this balances speed vs.
quality).",
      "Close any large files or unnecessary open editors in Cursor if you
don't need them, to limit what the agent considers. The agent tends to focus
on open files and recent context, so keeping only relevant files open can
improve response relevance [76] (Cursor Docs, 2023).",
      "Regularly refine and trim your Cursor project rules. Remove any
outdated or irrelevant rules that might be adding noise. Keeping rules
```

```
concise and up-to-date helps the model focus on the truly important guidance [22] (Cursor Docs, 2023).",

"If Cursor's responses slow down or you experience UI lag, try clearing the conversation history (start a new chat) because very long chat histories can bog down the model with too much context.",
      "Leverage Background Agents (if available) for long-running tasks like indexing or running tests, so your main session remains responsive. This way, a background process can handle heavy work and update you when done [77] (Cursor Docs, 2023).",
      "Keep your environment (Cursor app, VS Code, etc.) updated and monitor the Cursor Discord/forum for performance tweaks. The developers often optimize models and memory usage in updates, so staying current can yield speed improvements."
    ]
  },
  "artifacts": {
    "cursorrules_template": "# .cursorrules (template)\n- Objective: Keep edits minimal and repo-aware.\n- Strategy: SCoT (Signature → Constraints → Outline → Tests) then Apply.\n- Guardrails: Never modify files outside provided context list.\n- Output: Diffs only; no prose unless requested.\n- Tests: If code changed → add/adjust unit tests.\n- Review: Emit CHECKLIST before applying changes.",
    "prompt_templates_md": "## Prompt Templates\n### Feature (Plan → Apply)\n1) Plan: Summarize goal, files, functions, risks, tests.\n2) Apply: Produce minimal diff per file.\n### Refactor (Safe-Edit)\n- Keep public API stable; update imports/usages; run type/lint hints.\n### Bugfix (Repro → Fix → Test)\n- Reproduce with failing test; implement fix; make test pass.\n### TDD (Test First)\n- Propose tests; on approval, implement code to pass tests."
  }
}
```

---

[1] [14] [15] [16] [17] [18] [19] [26] How to build your agent: 11 prompting techniques for better AI agents - Augment Code
https://www.augmentcode.com/blog/how-to-build-your-agent-11-prompting-techniques-for-better-ai-agents

[2] [22] [23] [25] Cursor – Rules
https://docs.cursor.com/en/context/rules

[3] [28] [31] [32] [38] [39] [40] [48] [76] Cursor – Working with Context
https://docs.cursor.com/en/guides/working-with-context

[4] [47] [49] [50] [52] Cursor – Codebase Indexing
https://docs.cursor.com/en/context/codebase-indexing

[5] [41] [42] [43] [44] [45] [46] [77] Cursor – Keyboard Shortcuts
https://docs.cursor.com/en/configuration/kbd

[6] [51] [70] [75] Best Practices: .cursorrules - How To - Cursor - Community Forum
https://forum.cursor.com/t/best-practices-cursorrules/41775

7 30 55 56 57 58 72 Applying changes to a large file fails - Feedback - Cursor - Community Forum
https://forum.cursor.com/t/applying-changes-to-a-large-file-fails/47664

8 63 71 73 74 Cursor timeout agent mode - Discussions - Cursor - Community Forum
https://forum.cursor.com/t/cursor-timeout-agent-mode/45143

9 37 59 60 Grudgingly Accepting AI Coding Assistants | Stephen A. Fuqua - Blog
https://blog.safnet.com/2025/01/14/accepting-ai-assistants/

10 61 62 64 How Hidden Prompt Injections Can Hijack AI Code Assistants Like Cursor
https://hiddenlayer.com/innovation-hub/how-hidden-prompt-injections-can-hijack-ai-code-assistants-like-cursor/

11 20 21 27 36 Lead Agents with Prompts - DEV Community
https://dev.to/imaginex/lead-agents-with-prompts-2mpc

12 29 33 34 35 # Mastering Cursor AI: The Ultimate Guide for Developers (2025 Edition) - DEV Community
https://dev.to/mayank_tamrkar/-mastering-cursor-ai-the-ultimate-guide-for-developers-2025-edition-2ihh

13 24 65 66 67 68 69 A collection of prompts for generating high quality code... : r/ChatGPTCoding
https://www.reddit.com/r/ChatGPTCoding/comments/1f51y8s/a_collection_of_prompts_for_generating_high/

53 54 Read files in chunks of up to 250 lines - Bug Reports - Cursor - Community Forum
https://forum.cursor.com/t/read-files-in-chunks-of-up-to-250-lines/52597