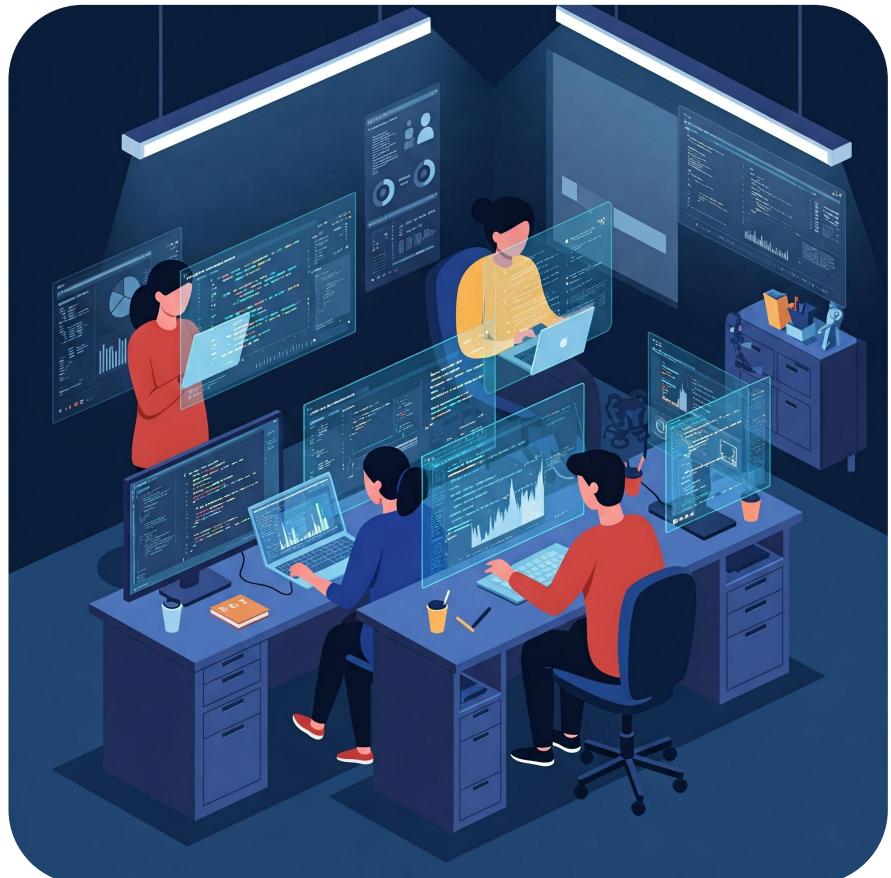


# AI-AIDED CODING



# Unlock Coding With AI

- AI tools can enhance the coding process significantly.
- This presentation explores AI's integration into coding practices.
- We will cover CLI and IDE solutions for AI coding.
- Practical experiences with different AI models shared today.



# Software Engineers Evolve Rapidly

- Software engineering roles will drastically change soon.
- Future CSE students will quickly adopt AI tools.
- Gain this essential knowledge to stay competitive.
- AI coding tools are becoming a vital asset.



# AI Coding Tools Explode Now

- AI coding tools evolved from annoyance to essential support recently.
- Tools like Claude, Aider, and Cursor emerged in recent years.
- Community innovations led before big model companies did.
- The software engineer's future is being redefined now.



# RECENT PAST

CLAUDE CODER FEBRUARY ~~2024~~ 2025

AIDER MAY 2023

CURSOR MID 2023

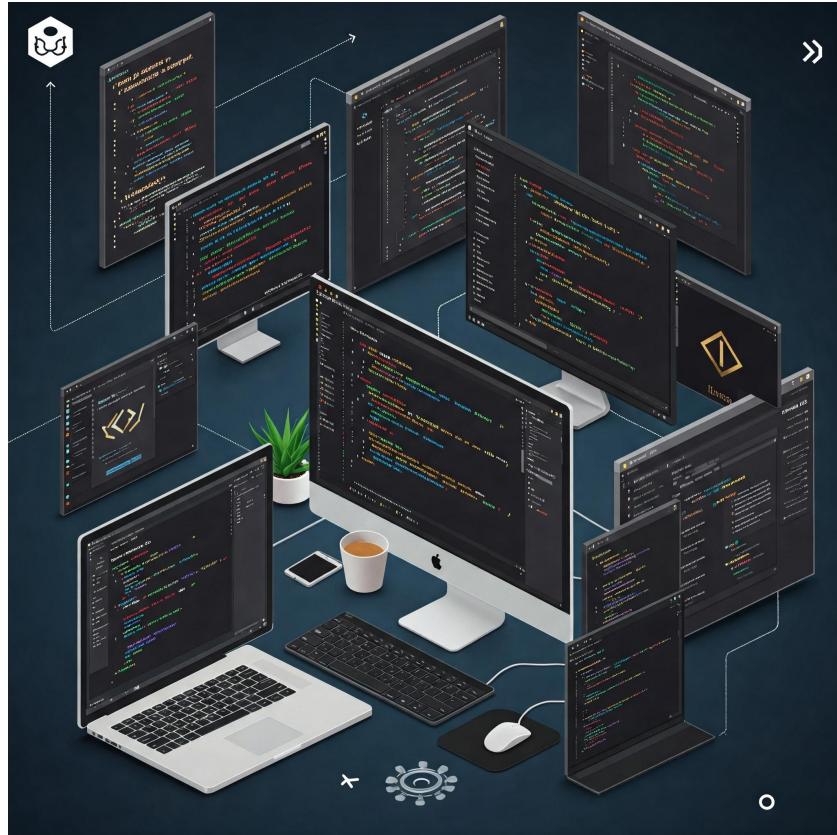
CODEX MAY ~~2023~~ 2025



THE BATTLE FOR THE SOFTWARE  
ENGINEER IS HAPPENING AS WE SPEAK!

# AI Coding Tool Landscape

- AI offers both CLI and IDE coding tools.
- Aider, Claude, and Codex are key CLI options.
- Gemini Code Assist is new to the toolset.
- Cursor and Windsurf are AI integrated IDEs.
- Jetbrains also offers AI coding features.



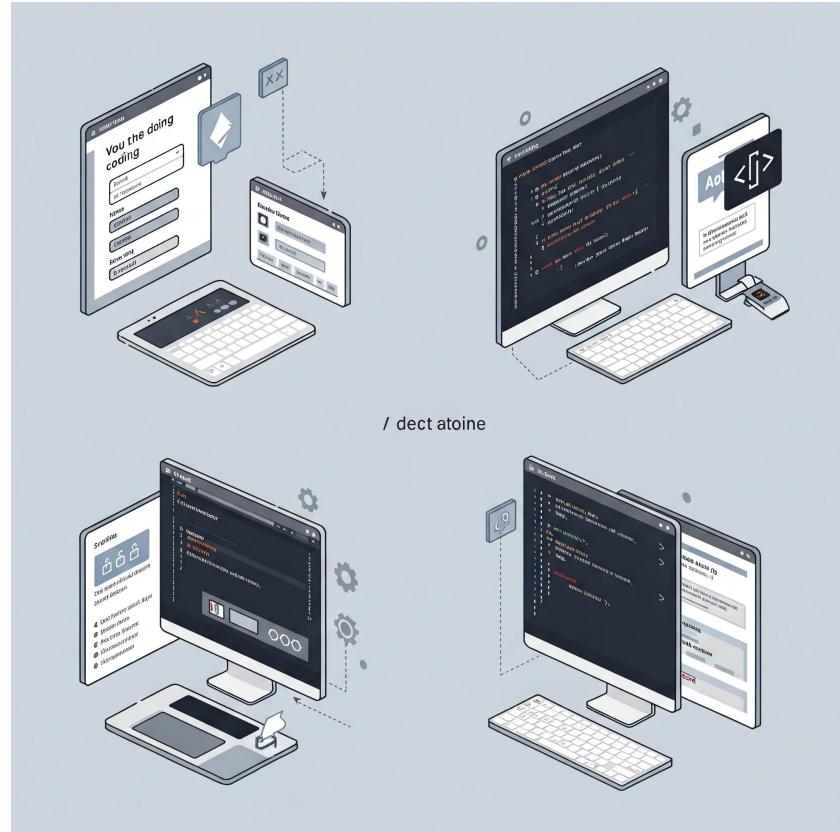
# Experiences With AI Coders

- AI coding tools like Claude Code show promising early results.
- Cursor is proving to be a very positive AI coding experience.
- Satisfaction with tools is related to the AI models used.
- Experiences with specific AI models will be discussed in detail.



# AI Coding Tools Comparison

- Aider needs mode switches via commands, learning is required.
- Claude Code responds to "do not code" requests directly.
- Cursor's ask mode tried to modify code anyway.
- Cursor requires clicks to apply code changes from chat.



# AI Coding Tools Comparison

```
~/Projects/frontend-stack-2 % aider
```



Mon May 26 20:04:14 2025

Aider v0.83.2

Main model: gemini/gemini-2.5-pro-preview-05-06 with diff-fenced edit format

Weak model: gemini/gemini-1.5-pro

Git repo: .git with 2,490 files

Warning: For large repos, consider using --subtree-only and .aiderignore

See: <https://aider.chat/docs/faq.html#can-i-use-aider-in-a-large-mono-repo>

Repo-map: using 4096 tokens, auto refresh

> /

/add	/context	/editor-model	/map	/read-only	/test
/architect	/copy	/exit	/map-refresh	/reasoning-effort	/think-tokens
/ask	/copy-context	/git	/model	/report	/tokens
/chat-mode	/diff	/help	/models	/reset	/undo
/clear	/drop	/lint	/multiline-mode	/run	/voice
/code	/edit	/load	/paste	/save	/weak-model
/commit	/editor	/ls	/quit	/settings	/web

# AI Coding Tools Comparison

\* Welcome to Claude Code!

/help for help, /status for your current setup

cwd: /Users/nlbruiv/Projects/frontend-stack-3

Tips for getting started:

1. Run /init to create a CLAUDE.md file with instructions for Claude
2. Run /terminal-setup to set up terminal integration
3. Use Claude to help with file analysis, editing, bash commands and git
4. Be as specific as you would with another engineer for the best results

Enrolled in Development Partner Program

> /init is analyzing your codebase...

- I'll analyze the codebase to create a comprehensive CLAUDE.md file. Let me start by exploring the key files and understanding the architecture.

- **Update Todos**

- Read main package.json and README files to understand project structure
  - Analyze storefront and graphql subdirectories for architecture
  - Check for existing development scripts and commands
  - Create CLAUDE.md with comprehensive guidance
  - Review Docker and deployment configurations

- **Update Todos**

- Read main package.json and README files to understand project structure
  - Analyze storefront and graphql subdirectories for architecture

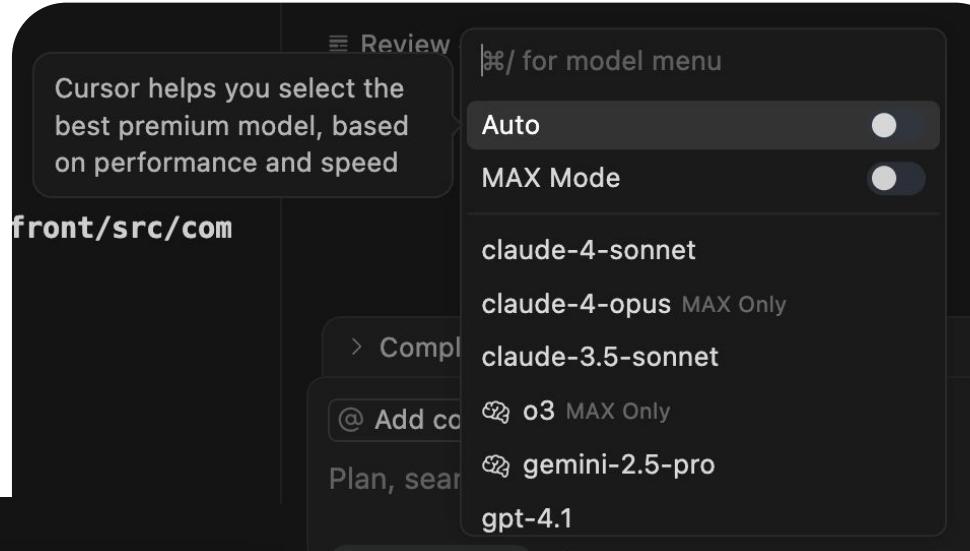
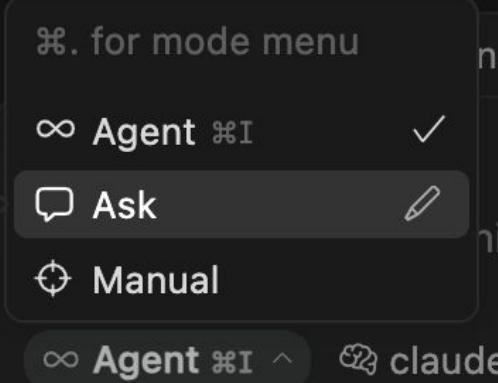
# AI Coding Tools Comparison

The screenshot shows a code editor interface with several panels:

- Left Panel:** A file tree for a project named "frontend-stack-2". The "productList" folder is expanded, showing files like "AlgoliaCore.tsx", "AlgoliaErrorBoundary.tsx", and "AlgoliaCore.d.ts".
- Central Editor Area:** The "AlgoliaCore.tsx" file is open. It contains TypeScript code for an "AlgoliaCore" component. The code includes logic for handling search results and configuration changes.
- Bottom Terminal:** A terminal window shows a GitHub commit message and a diff log. The commit message is about removing Algolia insights events. The diff log shows changes made to the "AlgoliaCore.tsx" file.
- Right Side Panels:**
  - Cursor Gitignore Configuration:** A panel showing a configuration snippet for the "InstantSearch" component. It includes a "key" prop that triggers a re-render when analytics settings change.
  - Solution Details:** A list of three steps to fix the analytics issue:
    - Forces a complete re-render of the `InstantSearch` component when either `shouldReportAnalytics` or `cookieAnalyticsApproval` changes
    - Ensures that the new analytics configuration is properly applied
    - Maintains the existing functionality while fixing the analytics issue
  - Review Changes:** A panel showing a completed change (1 file changed) and options to add context or plan/search/build anything.

# AI Coding Tools Comparison

Ask Cursor questions about your codebase



# Short Review of Several Tools

## Aider

Has a bit of a learning curve with modes

## Claude Code

Doesn't seem to need modes

## Cursor

in "ask" mode, outputs changes in chat  
instead of modifying code

# Claude Code: Terminal AI Delight

---

- Claude Code's interface and colors are visually appealing to me.
- Real-time feedback from Claude Code is highly beneficial.
- It integrates well within VS Code using the terminal.
- Requires explicit request to update memory file CLAUDE.md.
- Costs are visible within Claude Code, a closed source tool.



```
MISSING, please add tests for them to
storefront/src/utilities/ssr/index.test.ts
```

- I'll check the SSR utilities file and compare it with the existing test file to see what functions need tests.
- **Read(src/utilities/ssr/index.ts)...**
  - └ Read 313 lines (ctrl+r to expand)
- Now let me check the existing test file to see what's already tested:
- **Read(src/utilities/ssr/index.test.ts)...**
  - └ Read 519 lines (ctrl+r to expand)
- Now I can see which functions are missing from the tests. Let me compare:

#### Functions in index.ts:

1. ✓ isPDP - tested
2. ✗ isPLP - missing
3. ✗ isSearch - missing
4. ✓ isValidProductCode - tested
5. ✓ getProductCode - tested
6. ✓ getCategoryPathsFromPDPUrl - tested
7. ✓ isValidLocale - tested
8. ✗ isMatchLocaleCountry - missing
9. ✓ getCountryCodeFromAkamaiHeader - tested
10. ✓ getDefaultLocaleFromCountryCode - tested
11. ✓ isInvalidationPageTypeApplicable - tested
12. ✓ translateRegExp - tested
13. ✓ createCheckUnlinkable - tested
14. ✗ fetchQubitData - missing
15. ✗ getExperienceOptions - missing

# Claude Code

---

```
> /cost
└ Total cost: $3.15
  Total duration (API): 21m 30.1s
  Total duration (wall): 1h 25m 17.3s
  Total code changes: 785 lines added, 38 lines removed
  Token usage by model:
    claude-3-5-haiku: 103.1k input, 2.6k output, 0 cache
    read, 0 cache write
    claude-sonnet: 7.9k input, 25.3k output, 4.8m
    cache read, 631.4k cache write
```

```
> [ ]
```

```
? for shortcuts
```

```
> Can you add a jest test file next to
  storefront/src/utilities/parsers/getSiteTypeByHostname.ts to unit test its
  contents?

• I'll create a Jest test file for the getSiteTypeByHostname.ts utility. Let
  me first read the file to understand what needs to be tested.

● Read(storefront/src/utilities/parsers/getSiteTypeByHostname.ts)...
  └ Read 23 lines (ctrl+r to expand)

● Now let me check the constants to understand the hostname patterns:

● Read(storefront/src/utilities/constants/index.ts)...
  └ Read 73 lines (ctrl+r to expand)

● Now let me check the existing test file to understand the testing
  patterns:

● Read(storefront/src/utilities/parsers/index.test.ts)...
  └ Read 694 lines (ctrl+r to expand)

● Now I need to check the SiteType enum to understand the expected values:

● Search(pattern: "enum SiteType", path: "storefront/src", include: "*.ts")...
  └ Found 1 file (ctrl+r to expand)

● Read(storefront/src/utilities/graphql/codegen/index.ts)...
  └ Read 100 lines (ctrl+r to expand)

● Search(pattern: "SiteType\s*=\s*", path: "storefront/src", include: ...
  "*.ts")
  └ Found 0 files

● Bash(cd /Users/nlbruiv/Projects/frontend-stack-3/storefront && grep -n ...
  "SiteType" src/utilities/graphql/codegen/index.ts)
  └ 666: siteType: SiteType;
    673: siteType: SiteType;
    ... +95 lines (ctrl+r to expand)
  13564: $siteType: SiteType!
  13630: $siteType: SiteType!

● Read(src/utilities/graphql/codegen/index.ts)...
  └ Read 10 lines (ctrl+r to expand)

● Now I'll create the test file for getSiteTypeByHostname.ts:

● Write(file_path: src/utilities/parsers/getSiteTypeByHostname.test.ts)...
  └ Wrote 151 lines to src/utilities/parsers/getSiteTypeByHostname.test.ts
    import { getSiteTypeByHostname } from './getSiteTypeByHostname';
    import { SiteType } from '../graphql/codegen';
    describe('getSiteTypeByHostname', () => {
      describe('G-Star hostnames', () => {
        it('should return Gstar for www.g-star.com', () => {
          expect(getSiteTypeByHostname('www.g-star.com')).toBe(SiteType.Gstar);
        });
        it('should return Gstar for localhost', () => {
          ...
        });
      });
    });
  ...
  ... +141 lines (ctrl+r to expand)
```

# Claude Code

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS SEARCH

## ● Update Todos

- └ ~~Check current git status and working directory cleanliness~~
- └ ~~Read current version from VERSION file (create if missing)~~
- └ ~~Increment major version (X.Y.Z → X+1.0.0)~~
- └ ~~Update VERSION file with new version~~
- └ ~~Create git tag with v1version) format~~
- └ ~~Commit changes with proper message~~
- └ ~~Push both commit and tag to origin~~
- └ ~~Check for and update package.json if present~~

- Updated version: 0.0.1 → 1.0.0
- Created VERSION file
- Updated package.json
- Created git tag: v1.0.0
- Committed: "Release version 1.0.0"
- Pushed tag to origin

```
> /cost
  └ Total cost:           $0.1487
    Total duration (API): 1m 47.1s
    Total duration (wall): 3m 19.4s
    Total code changes:   3 lines added, 2 lines removed
    Token usage by model:
      claude-3-5-haiku: 4.8k input, 171 output, 0 cache read, 0 cache write
      claude-sonnet:    120 input, 3.7k output, 293.6k cache read, 10.1k cache write
```

> █

? for shortcuts

---

# Codex: “Cutting Edge” Dependencies



- Codex CLI requires NodeJS 22 for latest technology use.
- Codex CLI is model agnostic and uses "providers" like Aider.
- Haven't tried it (yet).

```
~/dev/github.com/openai/codex main  
openai > codex -m o4-mini
```

- OpenAI Codex (research preview) v0.1.04152057

```
localhost session: 63b06cac1dfb47b6b9e21c63b164bc9d  
↳ workdir: ~/dev/github.com/openai/codex  
↳ model: o4-mini  
↳ approval: suggest
```

**user**

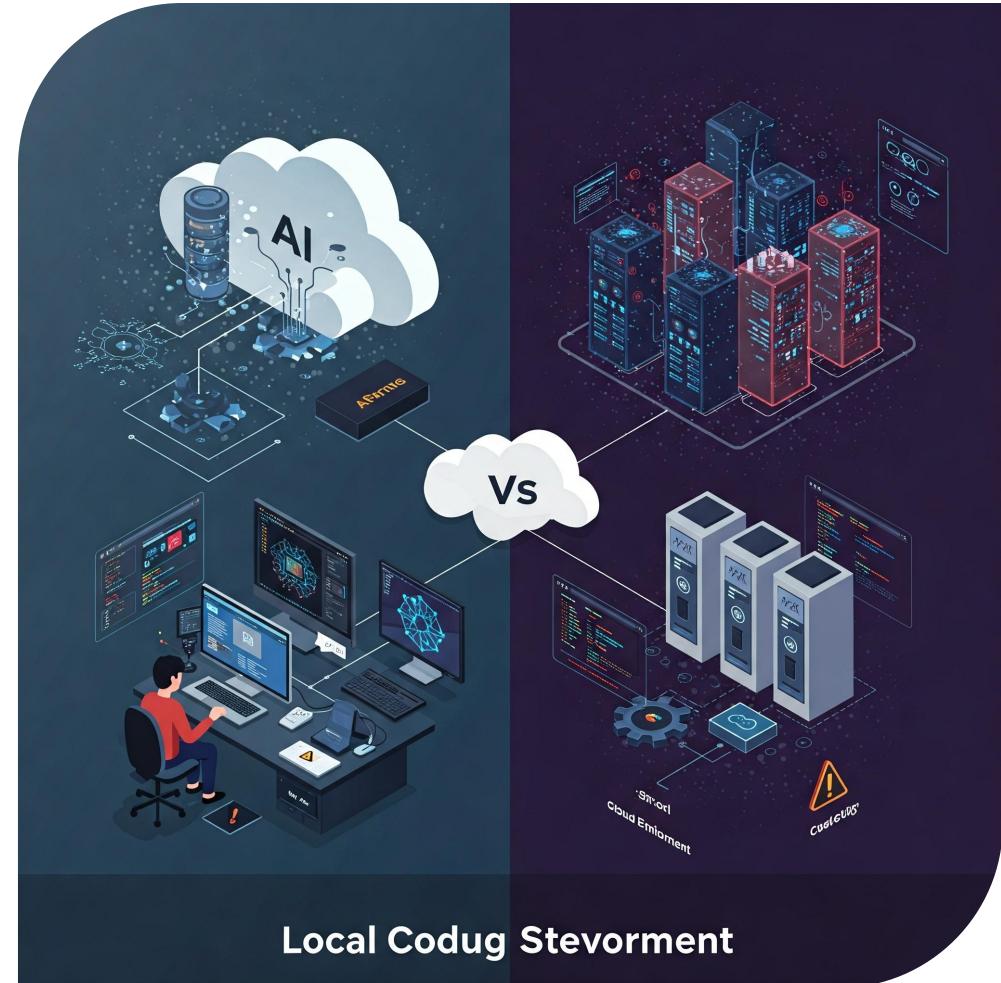
```
explain this codebase to me
```

```
( ●) Thinking
```

```
send q or ctrl+c to exit | send "/clear" to reset | send "/help" for commands | press enter to send
```

# Aider: Local Fail, Cloud Win

- Aider with Ollama and Qwen-3 was an attempt at local cheap development.
- Qwen-3 hallucinations occurred on my Mac Mini, potentially due to its size.
- Aider with Gemini worked well but lacked clear progress feedback.
- Aider is developer-focused with environment variables and manual commands.



## Hallucinations

Qwen3

> Can you extend '`emojiDictionary`' by adding 50 emoji as keys to this object and add as values an array with a minimum of 5 and a maximum of 40 words, verbs or nouns in English that could apply to the emoji?

First, I need to think of 50 emojis that aren't already in the dictionary. Let me check the existing keys. The current ones include things like 😊, 😋, 😌, 😍, 😎, 😏, 😐, 😑, 😒, 😓, 😔, 😕, 😖, 😗, 😘, 😙, 😚, etc. So maybe adding some that are less common or in other categories. For example, emojis related to technology, sports, or abstract concepts.



# Choose Your AI Coding Models

---

## Claude Sonnet 4

- **Claude-3.7-Sonnet** excels at coding, same as Gemini 2.5 Pro.
- ChatGPT-4o is also very good at coding tasks.
- Companies use proprietary models to learn company code styles.
- Amazon Q Developer is quite adept at Java coding tasks.
- JetBrains AI Assistant provides good Java coding support.

# Cost-Saving AI Coding Shortcuts

- Use chat AIs like ChatGPT for initial coding drafts.
- Generate project artifacts like README in AI chats.
- Transfer outputs from AI chat to IDEs easily.
- Start projects on AI mobile app, sync to desktop.
- Creates an index.html/js to quickly get going.



# HOW TO SAVE SOME COST

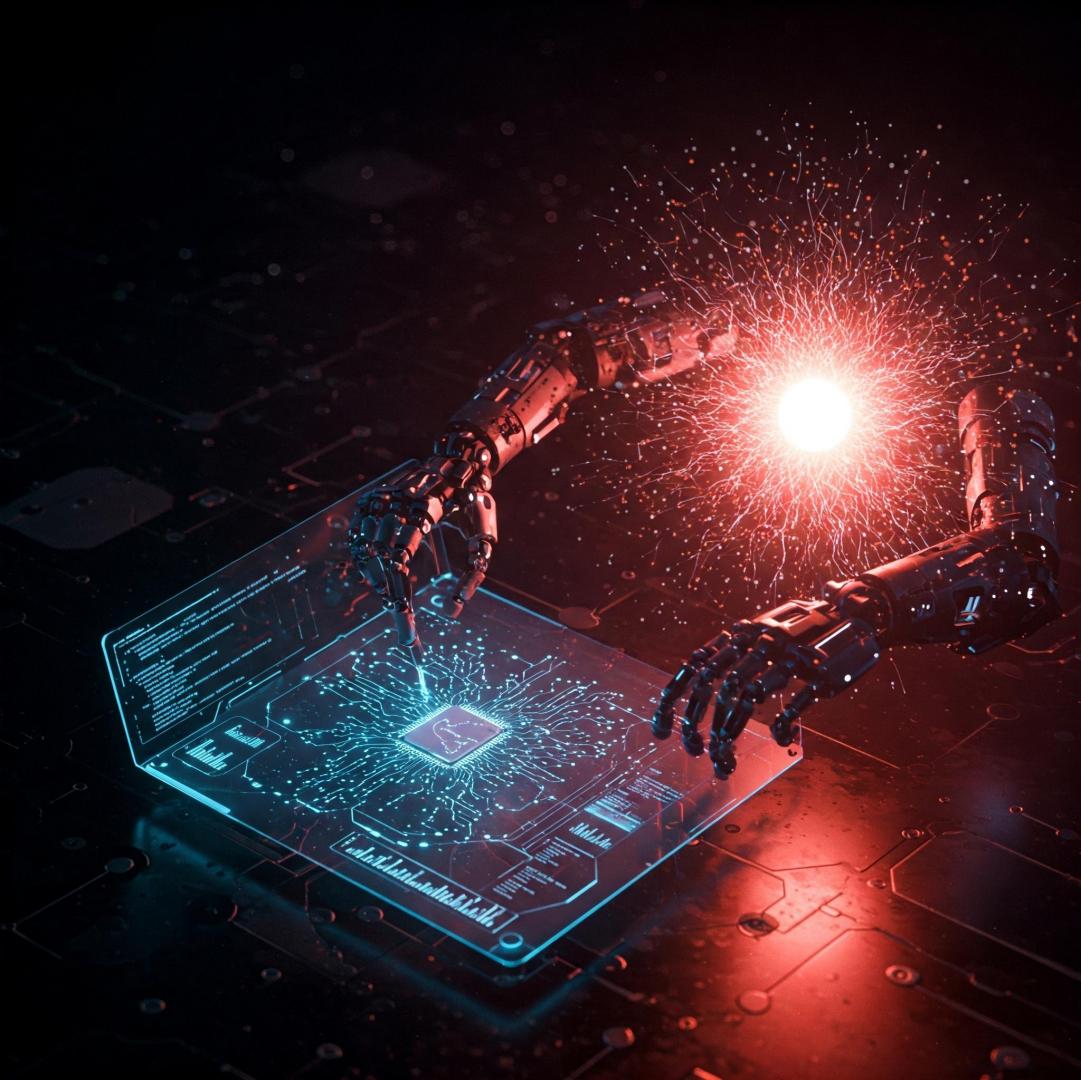
For smaller personal projects, start coding in ChatGPT, Gemini, or Claude...

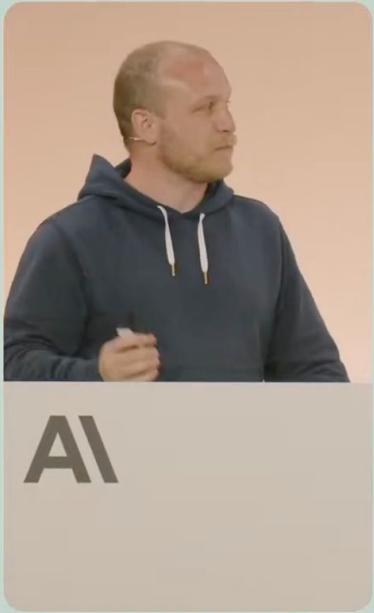


# AI Project Overview

## Limits

- Some AI tools build code layout maps for imports.
- Most AI assistants lack complete project detail overview for now.
- Open relevant files in Cursor for best results.
- Chat with AI on open files for issues.





## Use Claude Code to answer questions about your codebase

Example prompts:

- > How is `@RoutingController.py` used?
- > How do I make a new `@app/services/ValidationTemplateFactory`?
- > Why does `recoverFromException` take so many arguments? Look through git history to answer
- > Why did we fix issue #18363 by adding the if/else in `@src/login.ts` API?
- > In which version did we release the new `@api/ext/PreHooks.php` API?
- > Look at PR #9383, then carefully verify which app versions were impacted
- > What did I ship last week?

Code w/  
Claude



## Common workflows

Explore > plan > confirm >  
code > commit

> Figure out the root  
cause for issue #983,  
then propose a few  
fixes. Let me choose an  
approach before you  
code. ultrathink

Write tests > commit > code >  
iterate > commit

> Write tests for  
`@utils/markdown.ts` to  
make sure links render  
properly (note the tests  
won't pass yet, since  
links aren't yet  
implemented). Then  
commit. Then update the  
code to make the tests  
pass.

Write code > screenshot result  
> iterate

> Implement [mock.png].  
Then screenshot it with  
Puppeteer and iterate  
till it looks like the  
mock.

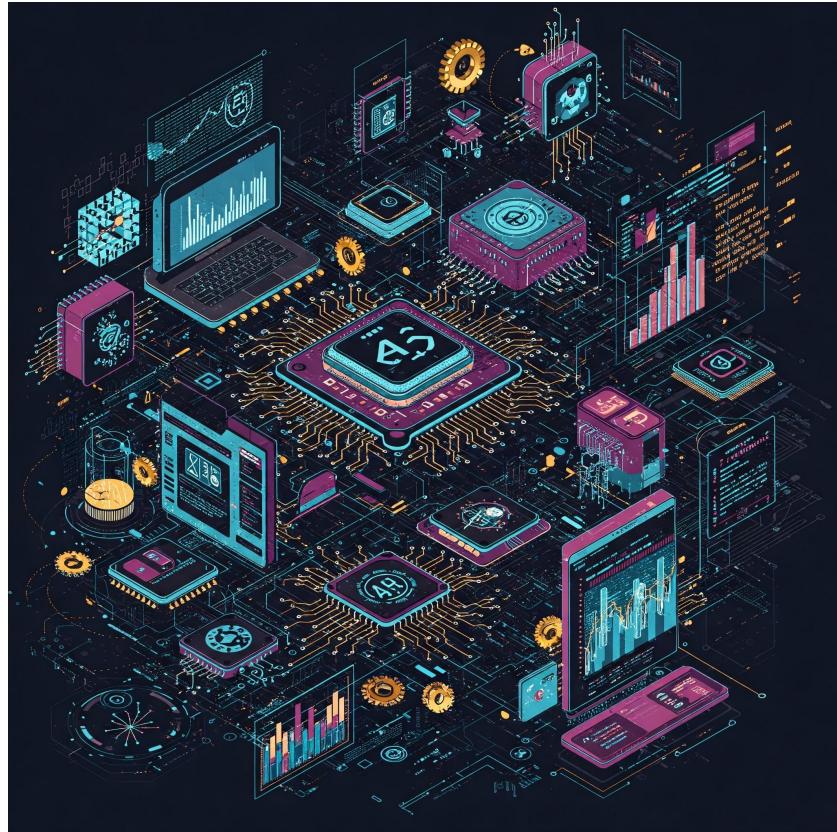
# AI TOOLS AND BIGGER PROJECTS

Do not (yet?) expect these tools to keep a full overview of all details in your project. But these tools usually make some 'mental' map of the layout of your codebase



# Multi-Model Aider: Select Models

- AI tools use various models for different tasks.
- Aider uses small models for commits and large ones for coding.
- Defining model usage is required within Aider.
- Some tools adapt model complexity based on needs.



# Multiple Models

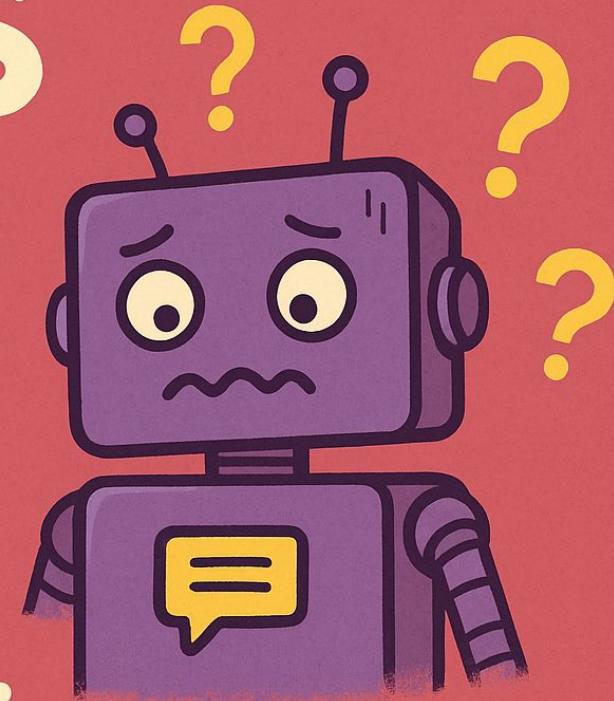
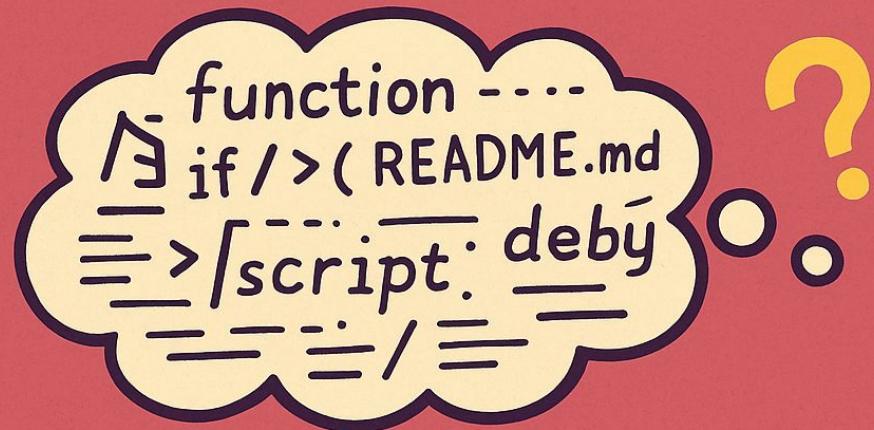


# AI Coding: Hallucinations & Precision

- AI models may generate incorrect code due to vague prompts.
  - Provide detailed context instead of generic instructions to AI.
  - AI can get stuck in loops creating endless files, debug scripts.
  - Knowing your codebase is key to giving precise AI instructions.



# PITFALLS



Hallucinations...

Be precise in what you want

# AI Coding Best Practices

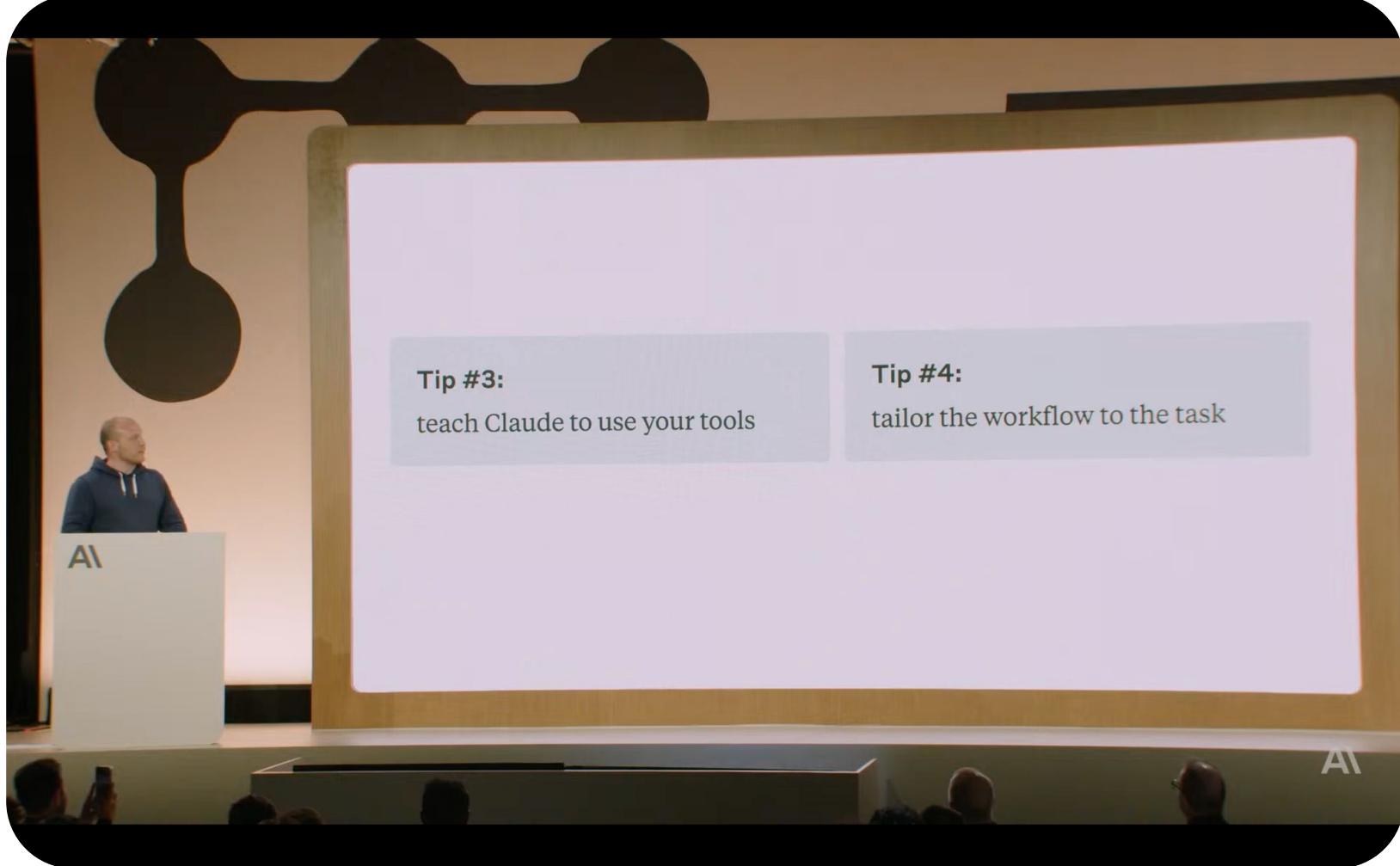
## Tip #1:

use codebase Q&A as a way to dip your feet into Claude Code

## Tip #2:

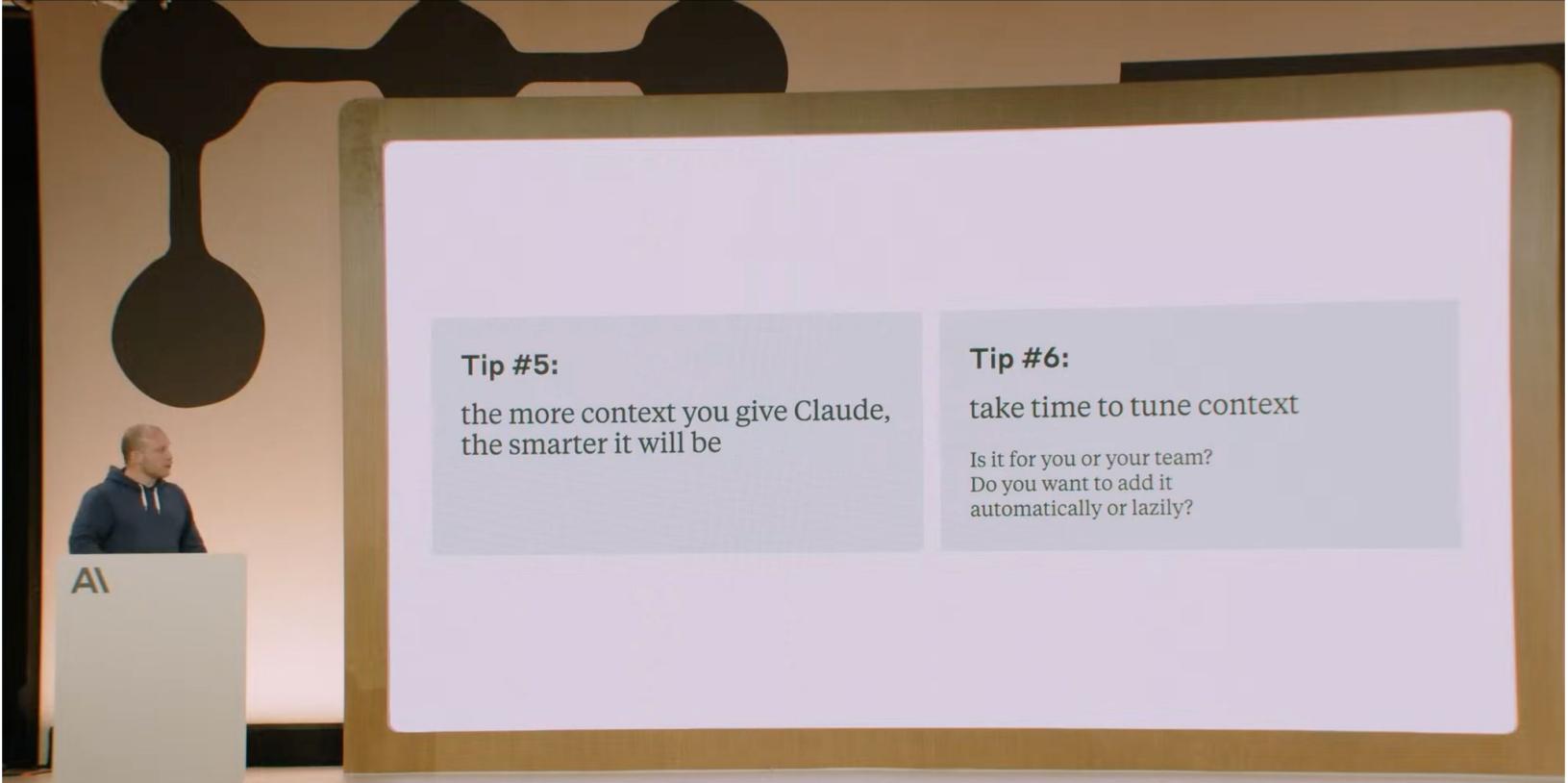
practice prompting, and start to understand what Claude Code “gets” immediately vs. what needs more specific instructions

- Commit code when work feels good enough.
- Aider does auto commits, which may feel controlling.
- Claude Code needs manual CLAUDE.md memory updates.
- Regularly update README.md



**Tip #3:**  
teach Claude to use your tools

**Tip #4:**  
tailor the workflow to the task



**Tip #5:**

the more context you give Claude,  
the smarter it will be

**Tip #6:**

take time to tune context

Is it for you or your team?  
Do you want to add it  
automatically or lazily?

# GENERAL ADVICE

DO COMMITS AS SOON AS YOU  
THINK THE WORK IS GOOD ENOUGH

WHEN USING CLAUDE CODE,  
SUGGEST NOW-AND-THEN TO LET  
IT UPDATE ITS MEMORY FILE

UPDATE YOUR README.MD  
WITH SOME FREQUENCY

# Review Before Accept: AI Code

- Review code**
- 
- Human PR reviews**
- 
- Do not blindly accept**  
**Accept**
- Scan for errors,**
- 
- Code review important**
- 

- Always review code changes before accepting from AI.
- Human PR reviews remain essential for software development.
- Do not blindly accept AI-generated code suggestions ever.
- Scan proposed modifications carefully for errors or issues.
- Importance of code review by humans is unchanged.

**NEVER BLINDLY  
ACCEPT CODE**

**ALWAYS READ/SCAN  
SUGGESTED CHANGES BEFORE  
ACCEPTING THEM**

**THE IMPORTANCE OF PR REVIEWS  
BY HUMANS REMAINS UNCHANGED**

# Getting Started: AI Coding

- Begin with Cursor or Windsurf IDEs.
- Try Claude Code within VS Code terminal.
- Experiment with multiple AI coding tools in parallel.



# HOW TO GET STARTED?



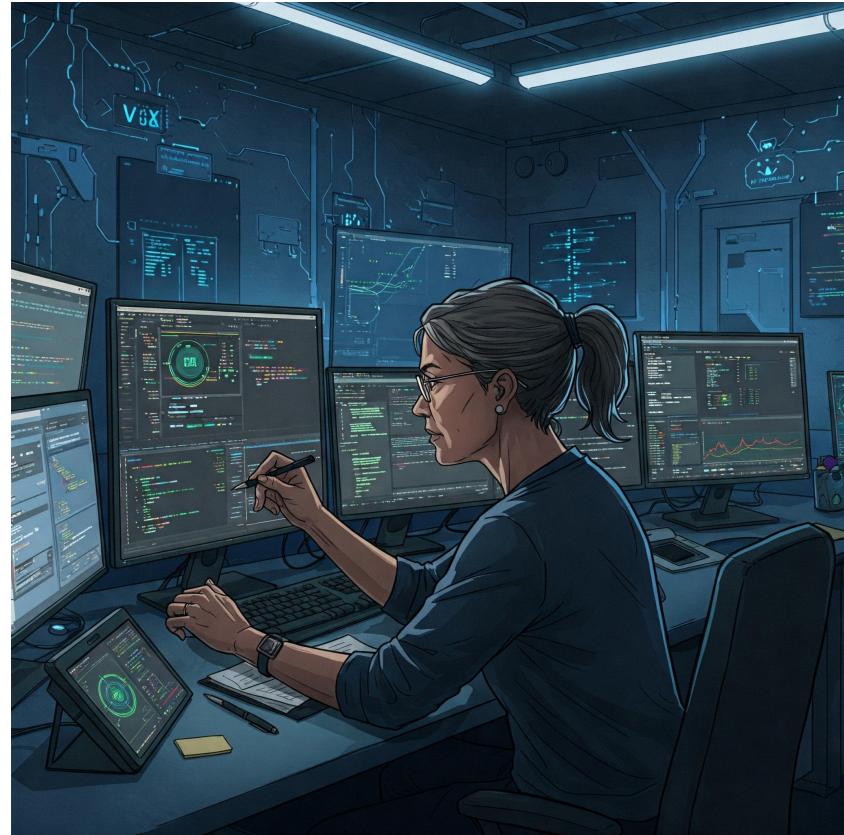
**Cursor or Windsurf**



## CODING IN STEREO

# Emerging AI Coding Tool Overload

- AI tool landscape is rapidly evolving with new releases.
- Google's Jules assists in updating GitHub repository packages.
- Keeping up with new AI coding tools is challenging task.
- High demand for Google's Jules tool outpaced initial supply.
- Emerging AI coding tools continually change development workflow.



# KEEPING TRACK

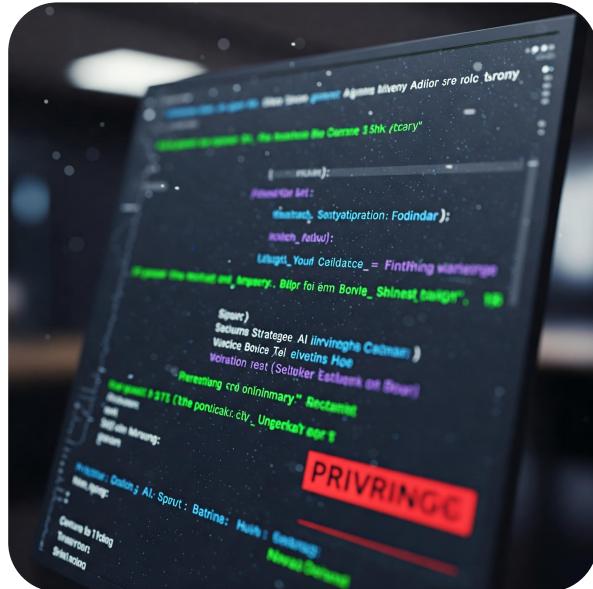


Jules



## MANY EMERGING TOOLS

# AI Coding: Secure, Smart, Strategic



- Maintain code privacy, avoid excessive context in public chats.
- Need to check which tools have what privacy conditions.
- Free AI chats often collect personal data.
- We're working on AI coding guidelines.

# NEXT STEPS



CODE  
PRIVACY

MULTI-MODEL  
APPROACH

# AI Coding: Key Takeaways

- AI tools are changing software engineering rapidly.
- Careful review of AI code is essential.
- Experiment with different AI coding tools.
- Balancing AI help and human expertise matters.
- Choosing the right AI model is crucial.



Claude 4 just refactored my entire codebase  
in one call.

25 tool invocations. 3,000+ new lines. 12  
brand new files.

It modularized everything. Broke up  
monoliths. Cleaned up spaghetti.

None of it worked.  
But boy was it beautiful.



A screenshot of a code editor showing a file named `MainTabletView.swift`. The code is AI-generated and appears to be a refactored version of an existing application's main tablet view. It includes imports for `UIKit`, `Foundation`, and `SwiftUI`, and defines a `MainTabletView` class that conforms to `View`. The class contains several methods and properties, including `init()`, `body`, and `onAppear`. The code uses various UI components like `Text`, `Image`, and `Text` with system images. There are also `log` statements and some Chinese text in the comments. The right side of the screen shows a sidebar with navigation links such as `CHAT`, `COMPOSER`, and `BUG FINDER`.

```
class MainTabletView: View {
    struct MainTabletView {
        @State private var selectedTab = 0
    }
    var body: some View {
        TabView(selection: $selectedTab) {
            Text("Home")
                .tabItem {
                    Label("Home", systemImage: "chart.bar.fill")
                }
                .log(0)
            Text("Transactions")
                .tabItem {
                    Label("Transactions", systemImage: "list.bullet.rtl")
                }
                .log(1)
            Text("Budget")
                .tabItem {
                    Label("Budget", systemImage: "plus.circle.fill")
                }
                .log(2)
            Text("Reports")
                .tabItem {
                    Label("Reports", systemImage: "chart.line.3d")
                }
                .log(3)
            Text("Settings")
                .tabItem {
                    Label("Settings", systemImage: "gear")
                }
                .log(4)
        }
    }
}
```

The code appears to be an accounting/finance tracking application with AI capabilities, allowing users to track expenses, income, set budgets, analyze spending, and generate reports. The AI has also added features like a budget management page and personal settings page.

I should check if there's already an existing codebase for this app that I need to modify or if I need to create the app from scratch. Let me explore the requirements to understand the current state.

Effects:

I'll help you implement this AI automatic accounting app based on the prototype designs you've provided. First, let me explore what we're working with in the requirements:

- 1. Home
- 2. Transactions
- 3. Budget
- 4. Reports
- 5. Settings

知乎·@无为设计研究所

# THANKS FOR LISTENING!

