# How to use AI to ship production code

# Brian Kelleher

Microdoc Founder and CEO

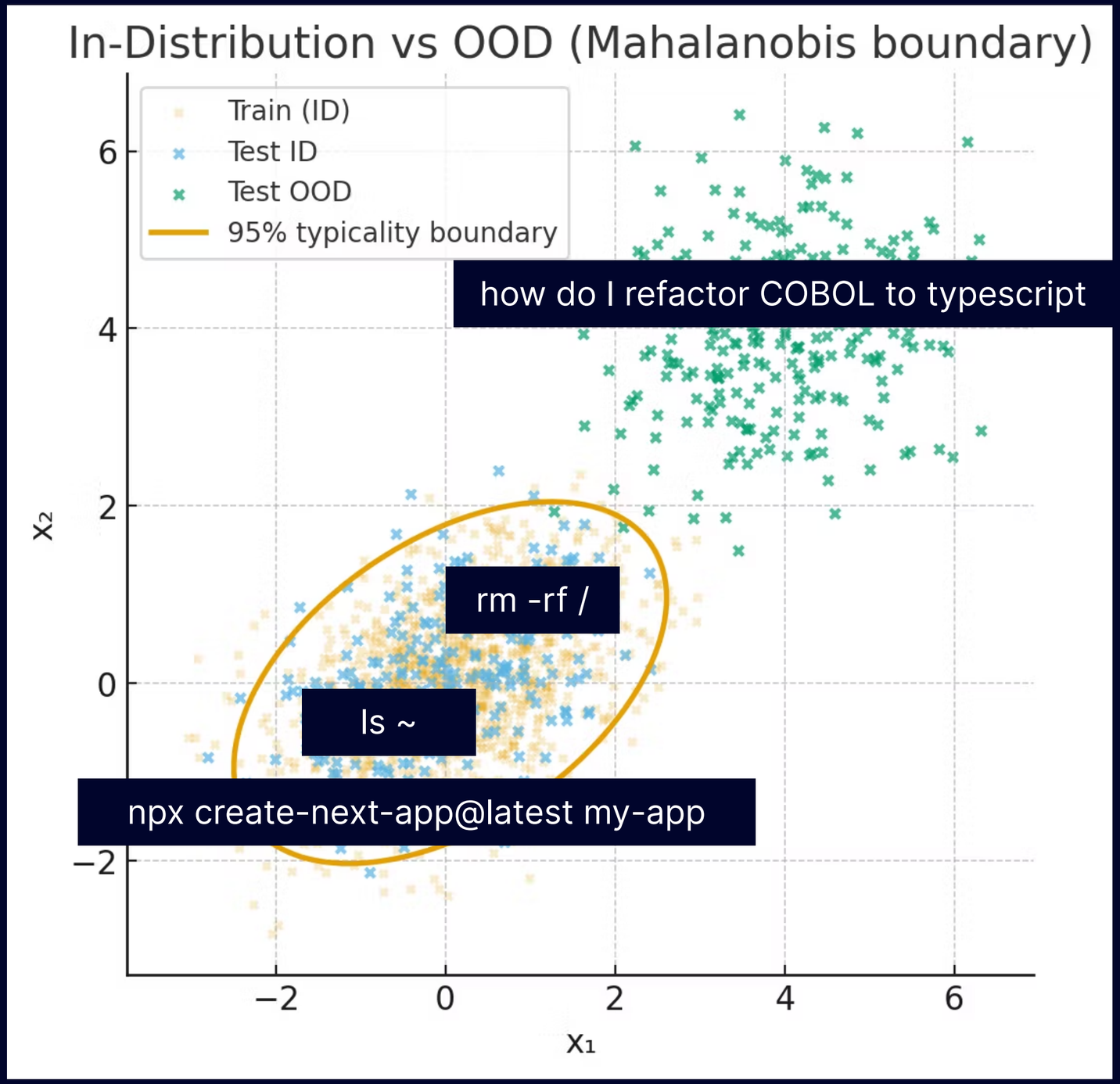
Edge City Fellowship

Using AI to prove novel mathematics

# Favourite AI coding tool?

**Supabase** ✓
@supabase

Supervising while Cursor deletes files at random


In-Distribution vs OOD (Mahalanobis boundary)
Train (ID)
Test ID
Test OOD
95% typicality boundary
how do I refactor COBOL to typescript
rm -rf /
ls ~
npx create-next-app@latest my-app
x₂
x₁
6
4
2
0
−2
−2
0
2
4
6

# Goal: Use AI to ship code to production

## Accuracy

Did the agent write valid code?

Did the agent understand what I wanted?

## Completeness

Did the agent finish the task in full?

Did the agent introduce regressions?

# Coding pipeline

1. **Understanding**
2. **Planning**
3. **Implementing**
4. **Checking**

1 Understanding

2 Planning

3 Implementing

4 Checking

# Practical prompting tips

Research in the codebase to find all relevant context.

Start by asking me a series of clarifying questions

Research in the codebase. Propose a few different options for the solution. Note which option you think is best

Don't write code yet, focus on understanding and analysing

1 **Understanding**

2 **Planning**

3 **Implementing**

4 **Checking**

🤖 ./setup-project <project-name>

README.md

USER_STORY.md

DESIGN.md

REQUIREMENTS.md

VALIDATION_STRATEGY.md

STATUS_UPDATES.md

PROGRESS_TRACKER.md

RETROSPECTIVE.md

scripts/README.md

1 **Understanding**

2 **Planning**

3 **Implementing**

4 **Checking**

The bash script is available on briankelleher.ie under the `projects` section

👨 Start work on Phase 1, ticking items off as you go

🤖 I've finished Phase 1

🔍 Checking in progress tracker...

🔁 Issues detected. Respawning agent...

1 Understanding

2 Planning

3 Implementing

4 Checking

🤖 I've finished

⚙️ Running static code analysis...

🛠️ Running tests...

🔁 Issues detected. Respawning agent...

1 Understanding

2 Planning

3 Implementing

4 Checking

## Prompting techniques

1. Research and understand thoroughly before writing any code
2. Don't write code yet, focus on researching, analysing, and understanding
3. Ultrathink - this is an extremely hard problem
4. Let's take a step back. Clearly lay out what you have done and what you have left to do
5. Ask me a series of clarifying questions before proceeding
6. Restate your understanding of this in your own words before proceeding
7. Implement the best, most long term, most elegant, most idiomatic solution

## Project and Workspace Setup

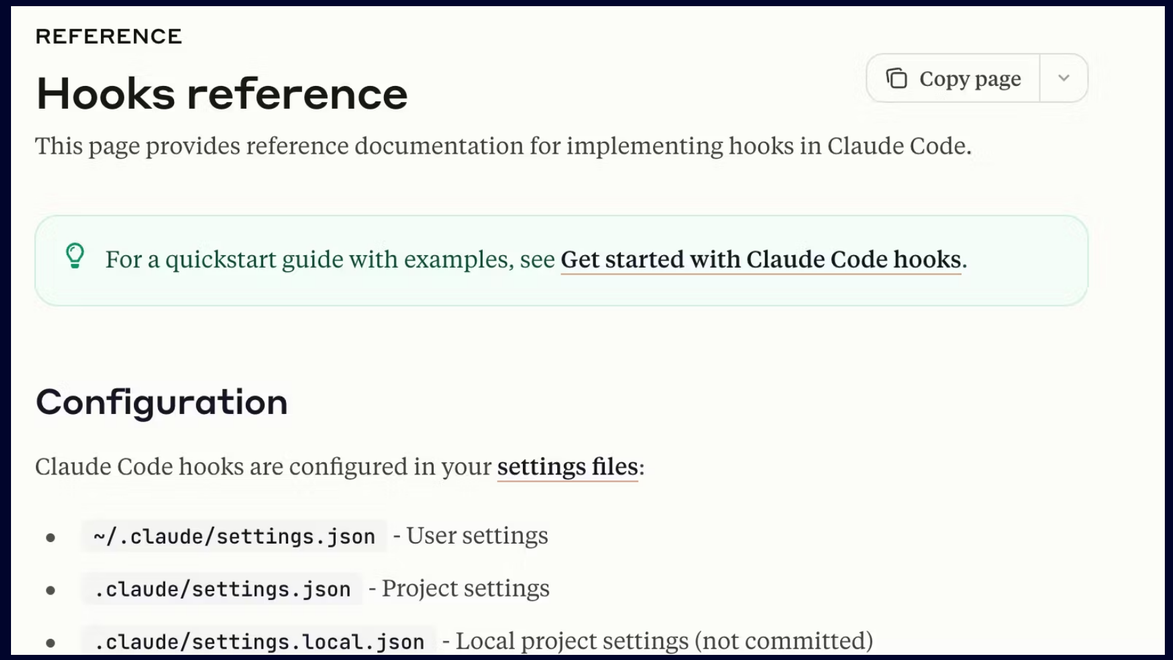
1. Bash-script-generated isolated workspaces
2. Use *AGENTS.md*, *CLAUDE.md*, *.cursorrules*, etc (config files)
3. Be opinionated when giving AI code style guidelines in config files
4. Continuously update config files as you discover limitations
5. Ensure proper CI, static type checking
6. Use git worktrees

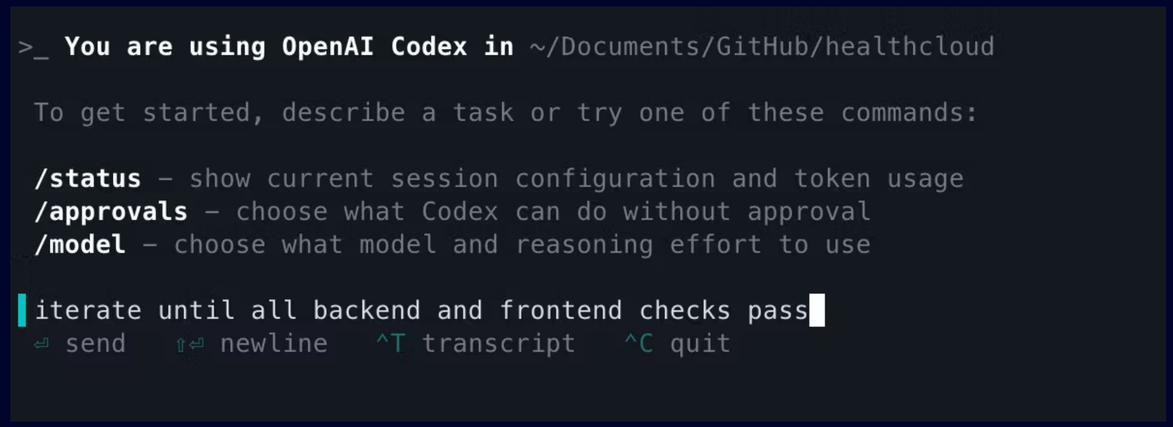
The bash script is available on briankelleher.ie under the `projects` section

Concrete tips for better AI coding

# Verification Techniques

1. LLM-to-LLM self-consistency checks.
   "My junior idiot colleague has done X, check their work'
2. Type-checking and static analysis and LSP in hooks 😍



# Logging

Add copious logs throughout in all relevant places

Create a script to verify and iterate until it passes

Concrete tips for better AI coding

# How to get better responses

### Compute

Ask the same prompt many many many times then get another model to choose the best response

### Orchestration

Split up a task over several sub-tasks

### Better models

Use better models (e.g. gpt-5-pro vs gpt-5)

### Better tools

You can use lots of compute getting an LLM to calculate 443×122, or just use a calculator tool

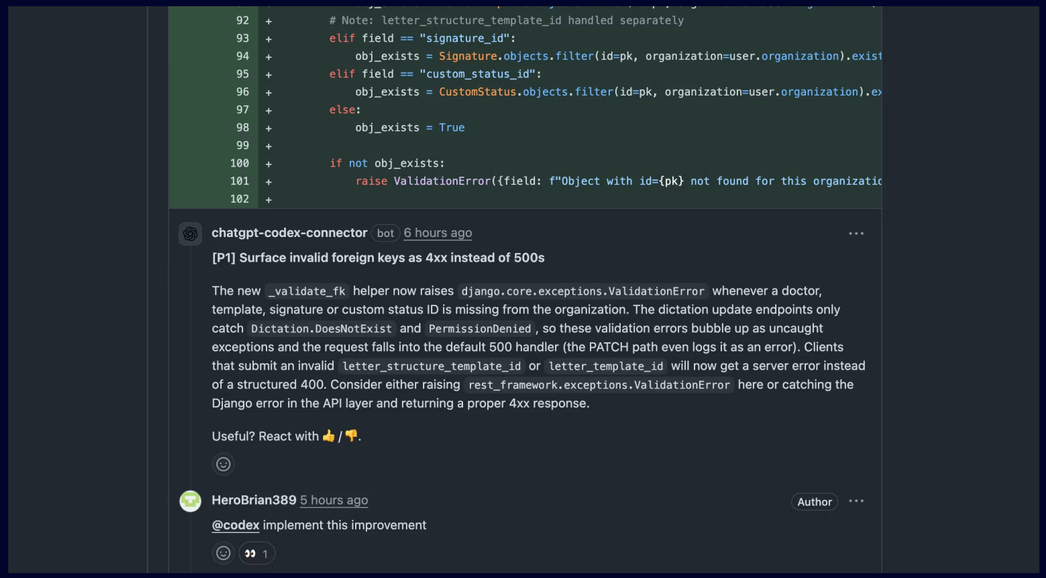
### Context engineering

Remove content that is not relevant to allow the LLM to focus on the most important parts of the question (Chekhov's gun)

Concrete tips for better AI coding

# AI code review

CodeRabbit



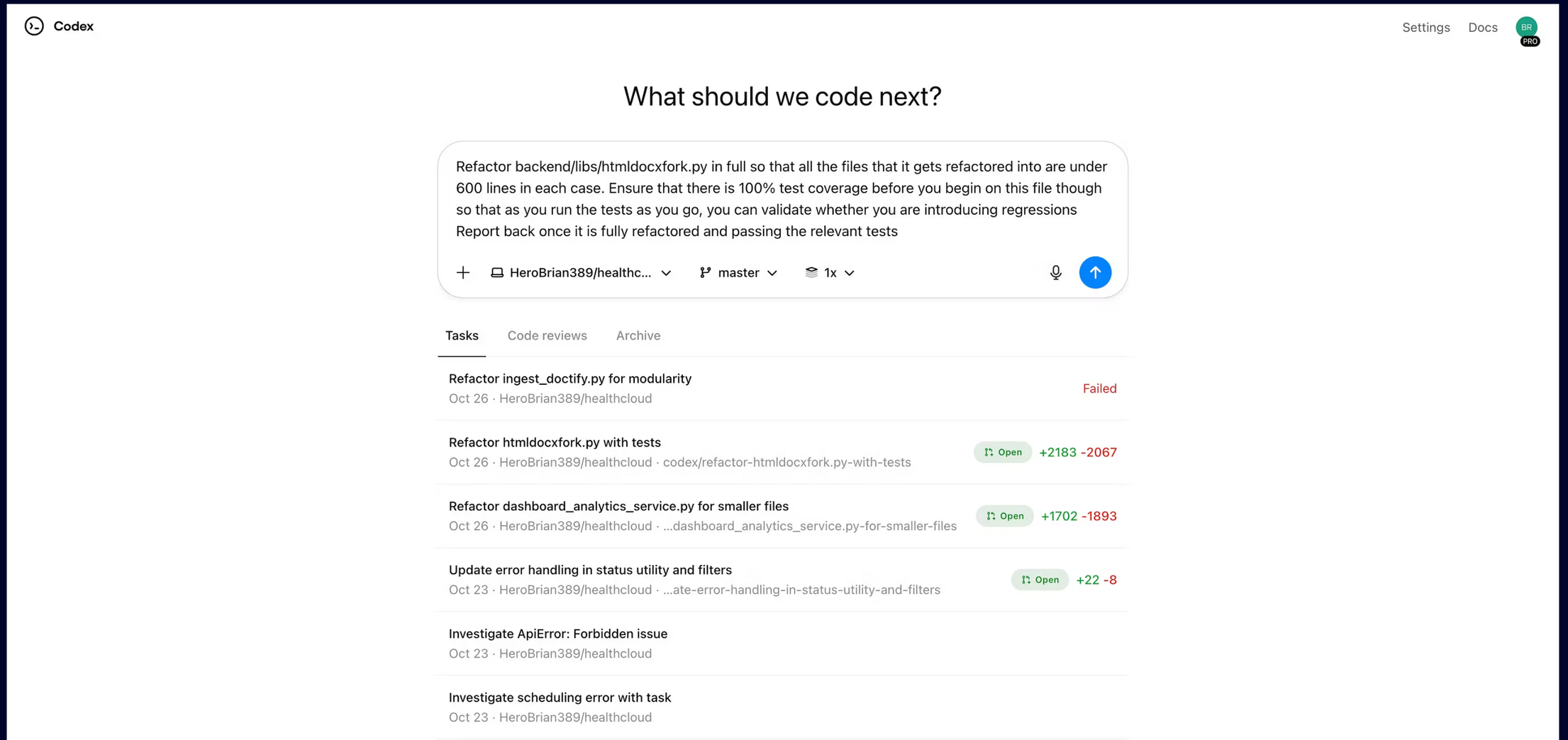
Concrete tips for better AI coding

# Using lightweight servers to remote AI code

Use `screen` or similar software to manage different pieces of work

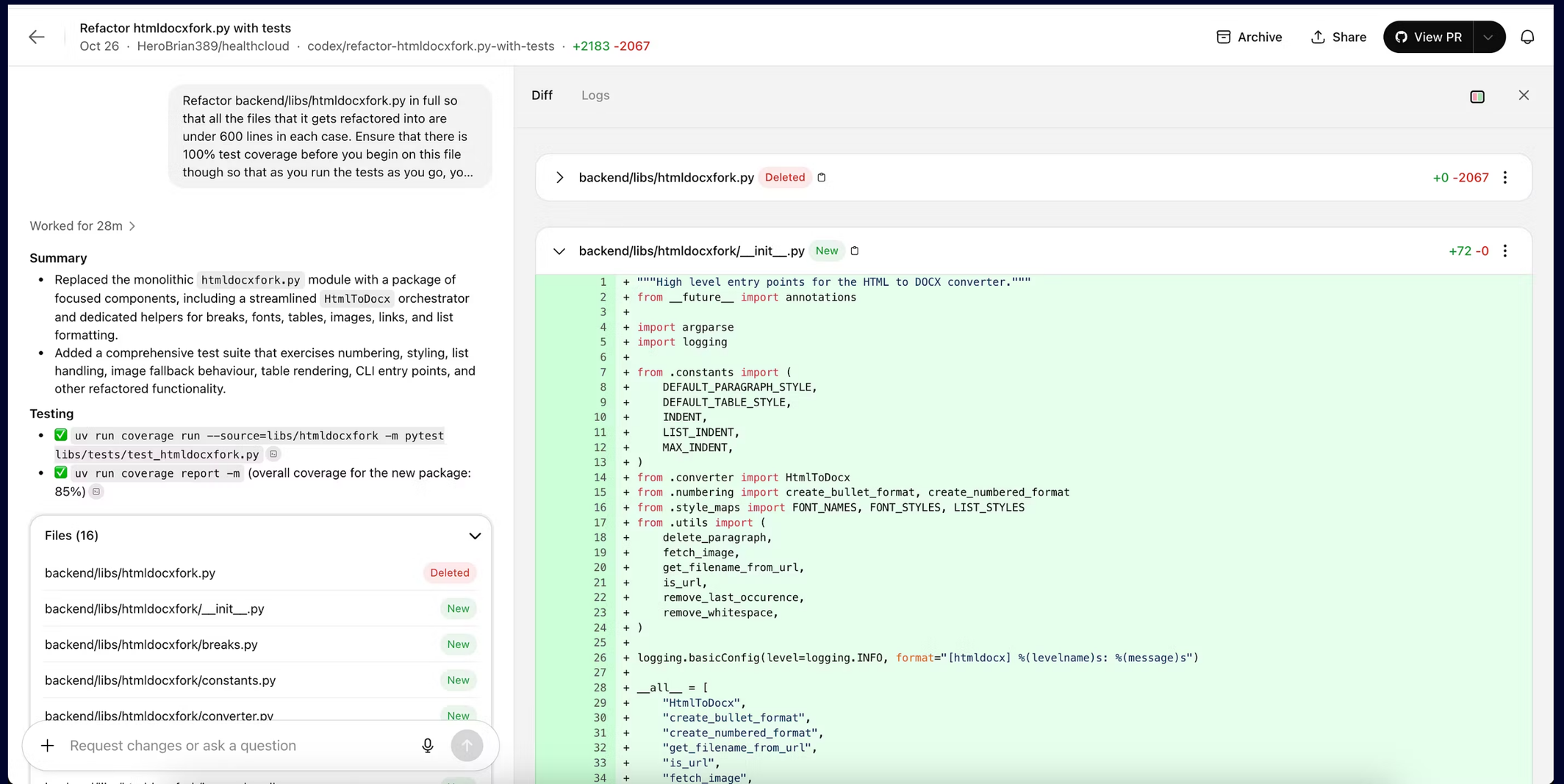
Concrete tips for better AI coding

# Using web-agents



Concrete tips for better AI coding

# Using web-agents



# Limitations

Memory

He's probably thinking about refactoring the whole codebase

Increase button size by 2px

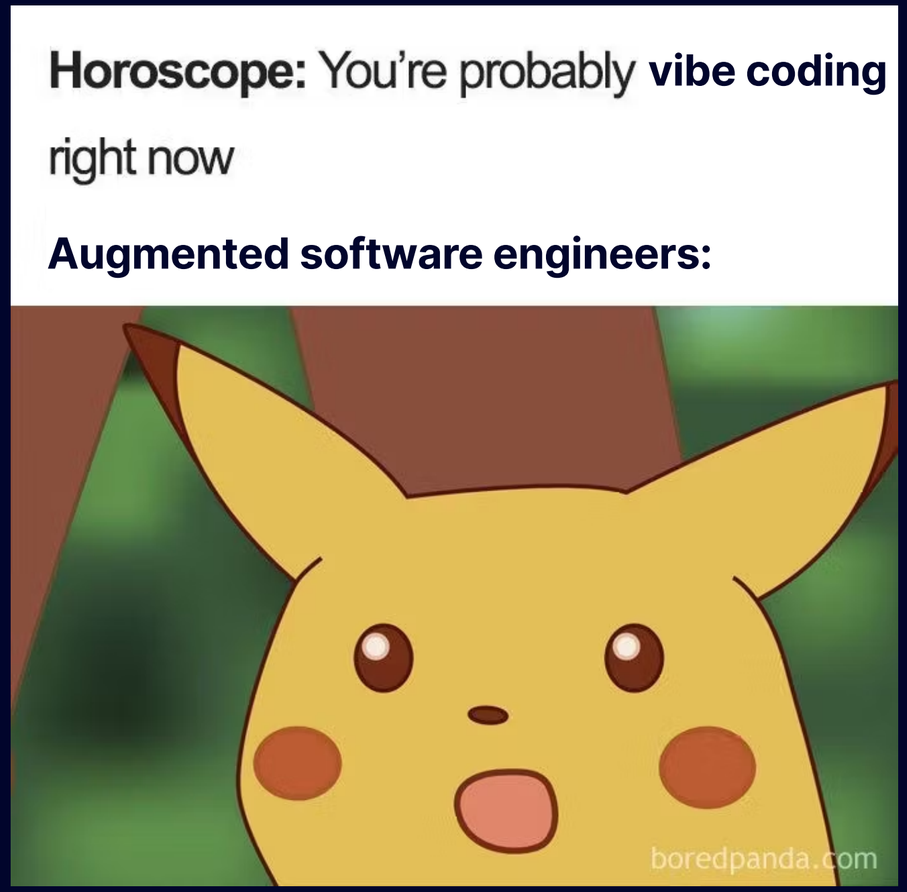

Epistemic access

## Predictions

1. Memory improvements
2. Less degenerate behaviour
3. Techniques will be productised
4. Increased model competition

**Best: GPT-5-codex-thinking-high**

Opus 4.2/4.5, Gemini 3 Pro coming soon



## Key takeaways

1. Always start with clarifying questions
2. Use structured projects to manage pieces of work
3. Use automated verification at every step

# Questions

briankelleher.ie/projects for technical writeups

brian@microdoc.io

Telegram: @briankelleher

Founder & CEO Microdoc

# Demo

Pitch

# Want to make a presentation like this one?

Start with a fully customizable template, create a beautiful deck in minutes, then easily share it with anyone.

Create a presentation (It's free)