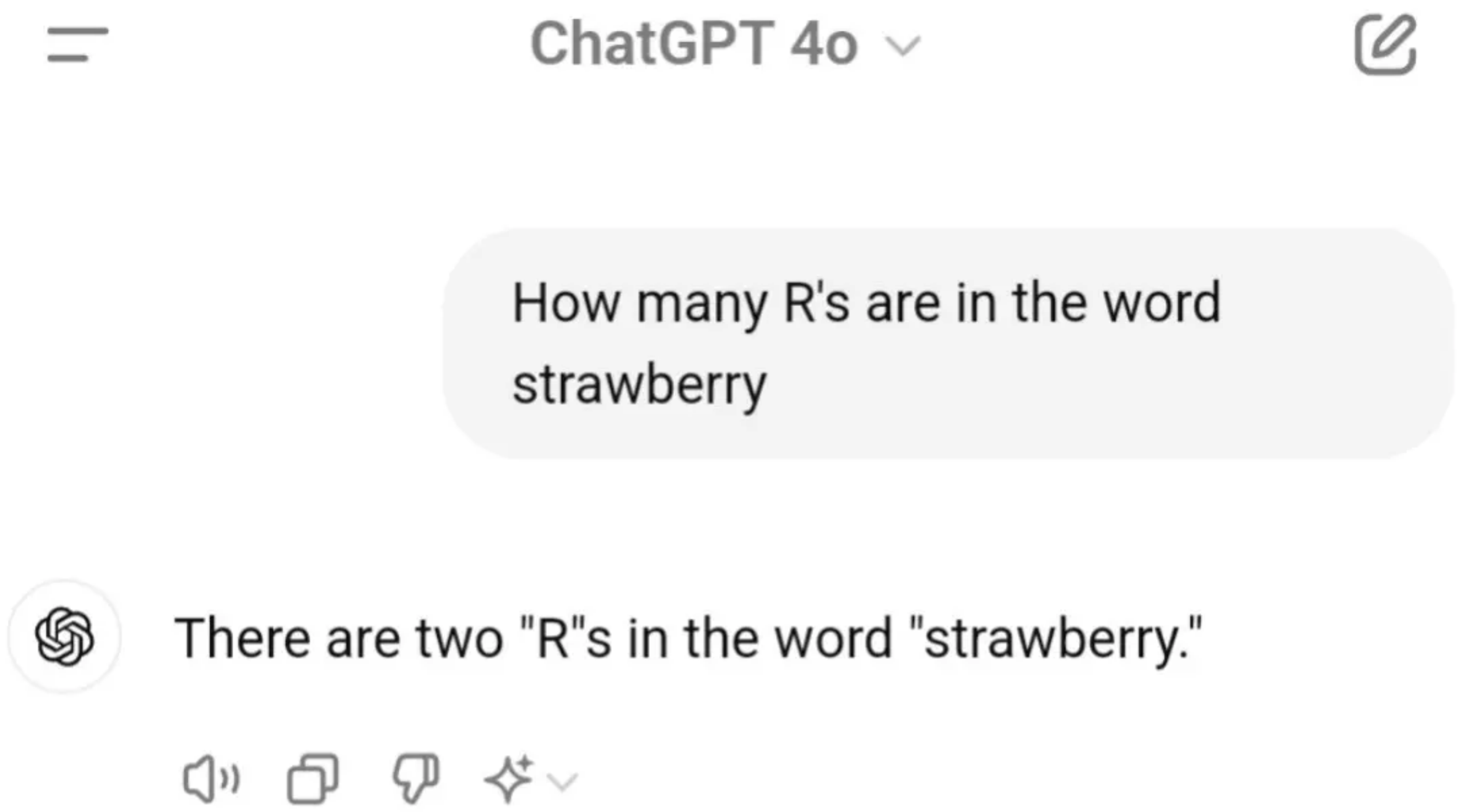
Hello 👋

Vibe Coding a Home Server on the Raspbery(*) Pi
What AI-Native Development Actually Looks Like

Wim De Troyer - BeJUG/BruJUG


ChatGPT 4o

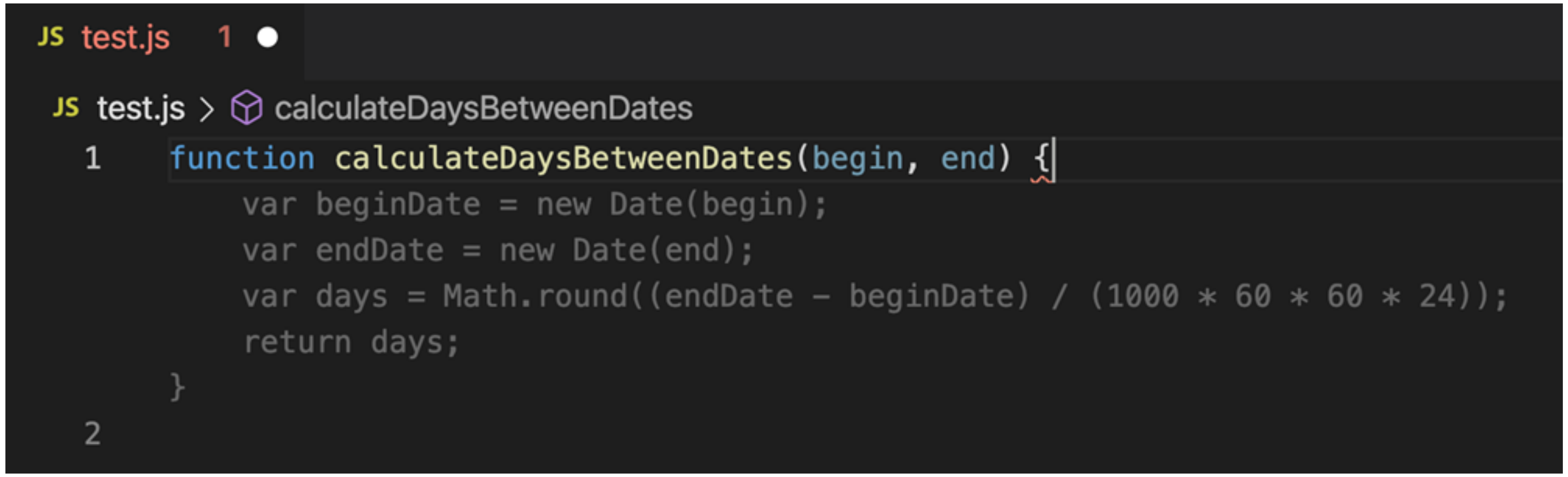
How many R's are in the word strawberry

There are two "R"s in the word "strawberry."

First a bit of LLM history…

A bit of LLM history…

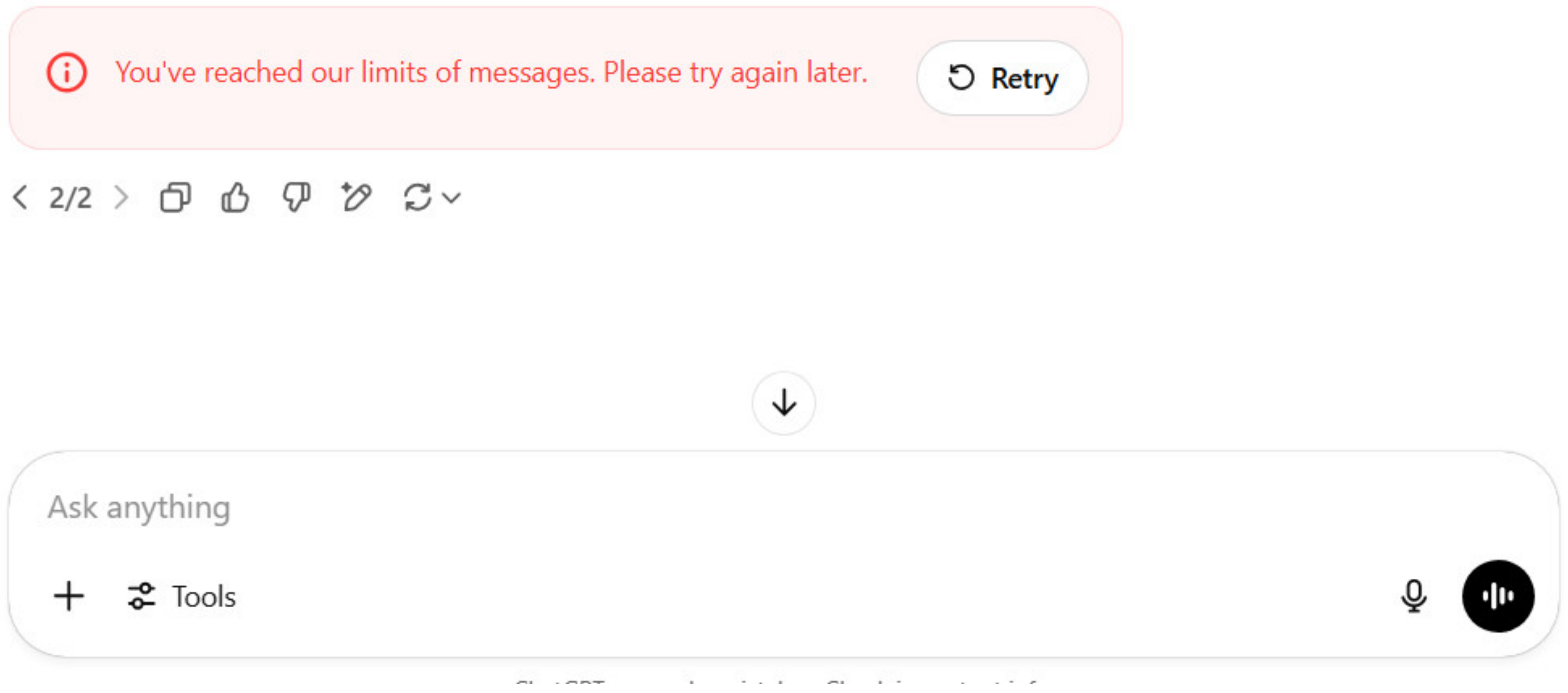
Github copilot - 2021



```javascript
function calculateDaysBetweenDates(begin, end) {
    var beginDate = new Date(begin);
    var endDate = new Date(end);
    var days = Math.round((endDate - beginDate) / (1000 * 60 * 60 * 24));
    return days;
}
```

A bit of LLM history…

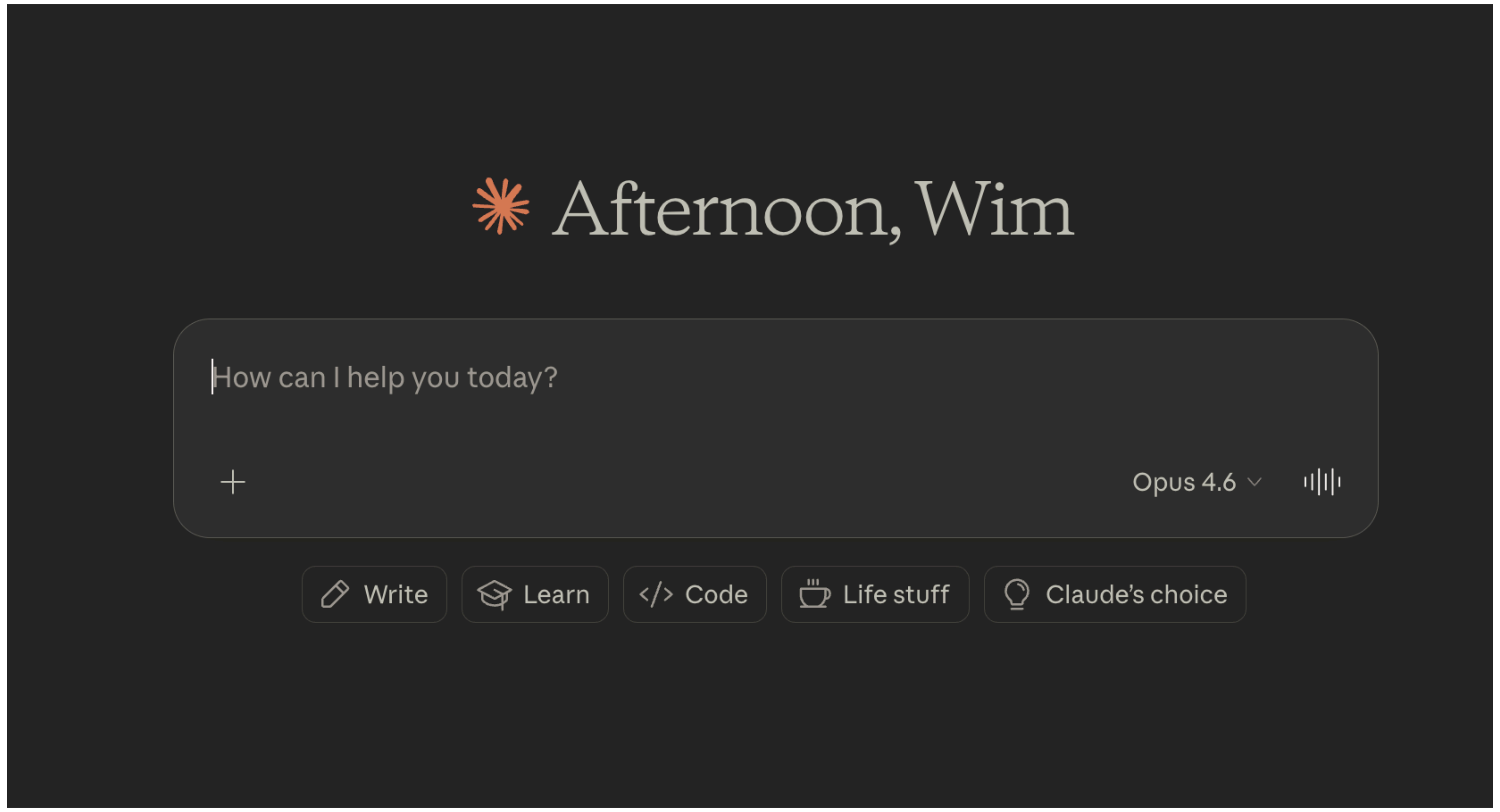
ChatGPT - 2022



You've reached our limits of messages. Please try again later.

A bit of LLM history…

Claude - 2023

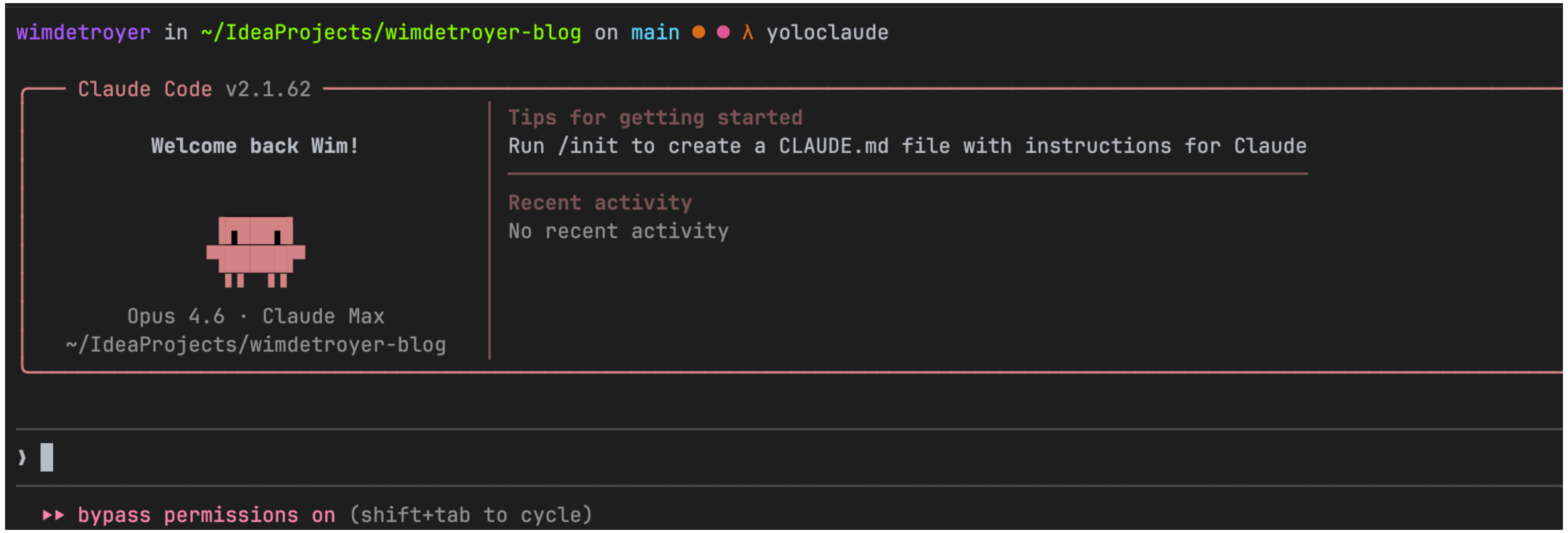

Afternoon, Wim

How can I help you today?

Opus 4.6

A bit of LLM history…

Claude Code - 2025 - “Agentic AI”



```
wimdetroyer in ~/IdeaProjects/wimdetroyer-blog on main ● ● λ yoloclaude

── Claude Code v2.1.62 ──────────────────────────────────────────────
                                    │ Tips for getting started
          Welcome back Wim!         │ Run /init to create a CLAUDE.md file with instructions for Claude
                                    │ ─────────────────────────────────────────────────────────────────
                                    │ Recent activity
                                    │ No recent activity

       Opus 4.6 · Claude Max
  ~/IdeaProjects/wimdetroyer-blog
─────────────────────────────────────────────────────────────────────

❯ 

 ▸▸ bypass permissions on (shift+tab to cycle)
```

This agentic thing seems like a big deal…

- Prompting no longer a nuisance
  - copy pasting snippets -> staying in the terminal
- Use a (unix) ‘dumb’ tool pipeline
- Starts off great… but converges towards error

But it got better and better…

- Models start shifting to these long, conversation style usages
  - Models stay on track even when context fills up
- UX improvements to increase productivity
  - MCPs: jira, AWS, playwright…
  - Context7: up-to-date documentation
  - Skills, …

2026: Peak of The Agentic AI Hype! Time for some wild claims!

“AI agents can handle long-running coding tasks mostly unsupervised”

“No need to pay for apps anymore, just roll out your own *bespoke* ones”

“Handwriting code is a thing of the past”

Am I missing out?

Lets put it to the test… However!

Can’t go all the way at work…

- Liable
- pesky guardrails / colleagues
- Tooling limitations
- Bounded by projects / enterprise imagination

The idea: A homeserver on the RPi

- Wanted to try out Raspberry Pi anyway
- Don’t look at code, just ‘vibe’ with Claude
  - See how far this’ll take me
- A way to validate all claims at once

Calorie tracker - Initial functional idea

- Calorie/macro tracker
  - AI input
  - trends
- Self-hosted, on the pi
- Accessible anywhere
- Secure
- Mobile-friendly

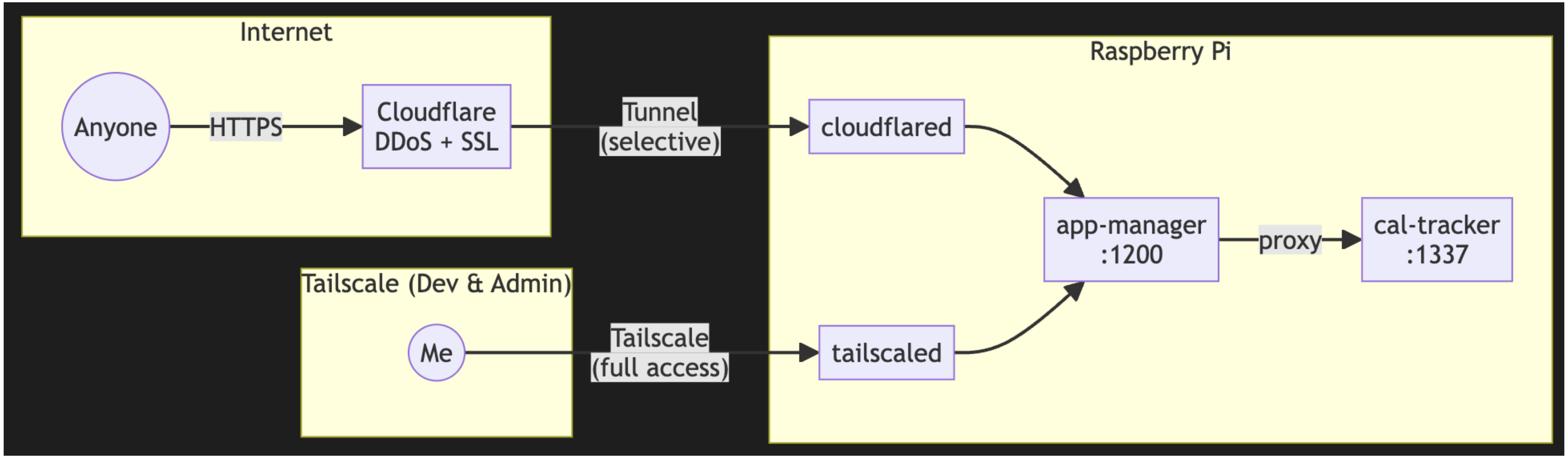
Technologies

- Java (ofcourse 😉)
- **Spring** and **HTMX**
- Safe, robust, simple technologies
- loads of LLM training data
- No build step for front
- I master these already (if things go wrong)

Way-of-working

- Solely used opus 4.6/4.5
- YOLO mode
- Heavily leverage plan-mode
  - But use it in **unstructured/stream-of-consciousness** way
  - Refine on this plan until satisfied
  - Accept plan, clear context
- Get a coffee, come back 20 minutes later
- Ran out of Claude Code tokens using the 20EUR/month subscription

Initial Architecture



This isn’t programming… or fun!

- Had no idea what the codebase was like
- Fixed the same bugs
- Progress seemed random

LLMs with no structure or architecture regress to the mean

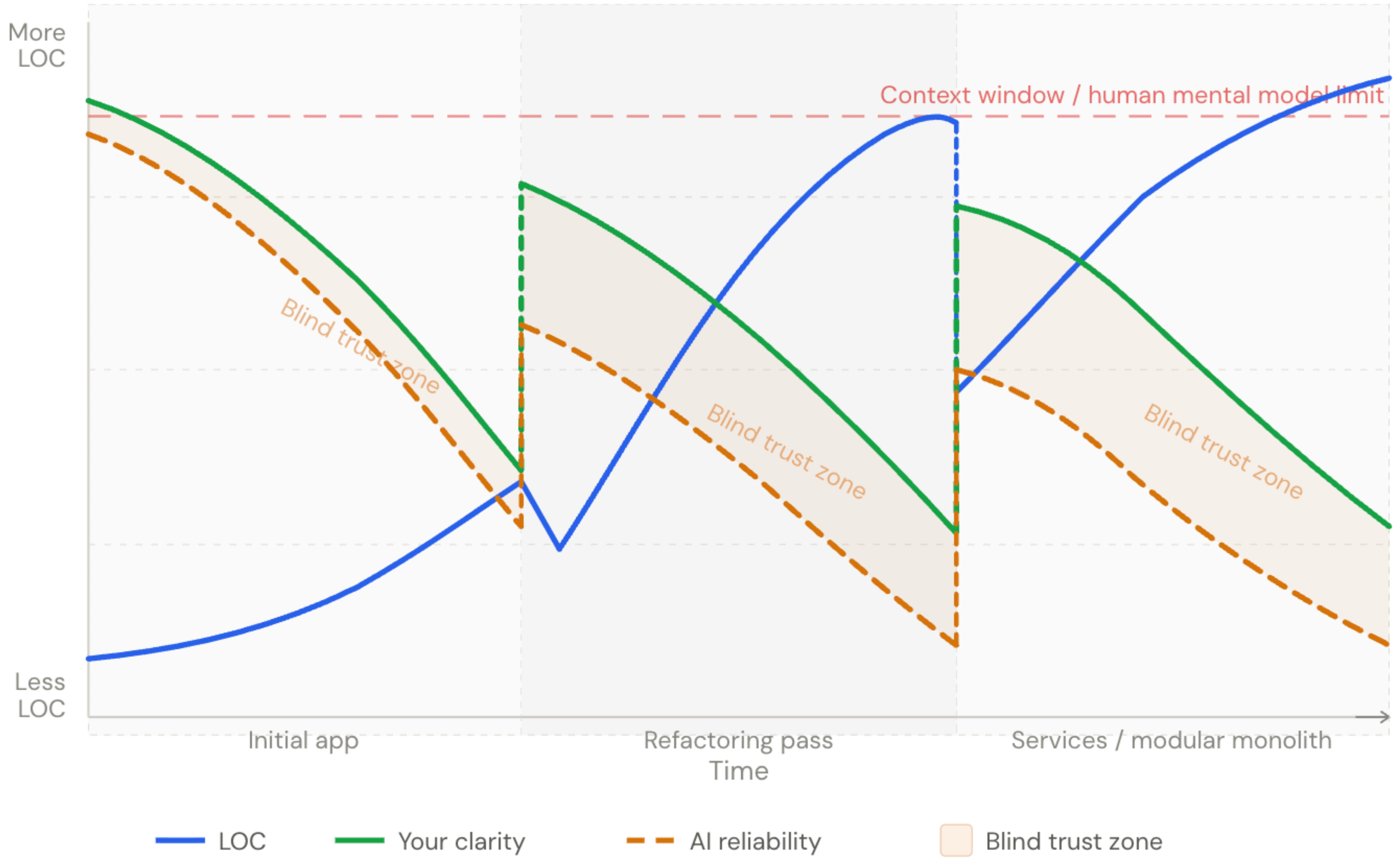
No (good) test suite means no self-correcting feedback loop

But first.. A demo of cal-track!

Putting a framework in place

Code architecture

- E2E playwright tests and unit tests
- Refactoring overcomplicated code
- Package-by-feature
  - Limits context of the LLM
- Prune dead code
  - “Adding code that never runs reduced bug-detection accuracy to 20.38%.”
- Template app with all (opinionated) best practices
- homeserver repo, Claude.md
  - small architectural pointers, gotchas, …
  - Don’t overdo it: code is the best reference and the rest will drift


More LOC
Less LOC
Context window / human mental model limit
Blind trust zone
Blind trust zone
Blind trust zone
Initial app
Refactoring pass
Services / modular monolith
Time
LOC
Your clarity
AI reliability
Blind trust zone

Sometimes called ‘harness engineering’ or ‘LLM guardrails’

But… these are just good software practices from before LLMs :-)

Putting a framework in place

Infrastructure

- app-manager as meta-layer
- Apps remain simple, just get registered
- Apps have spring-actuator exposed for lifecycle events
- Standardized deployment & profiles
- Linux FS conventions
- Structured logging

Things started to get out of hand…

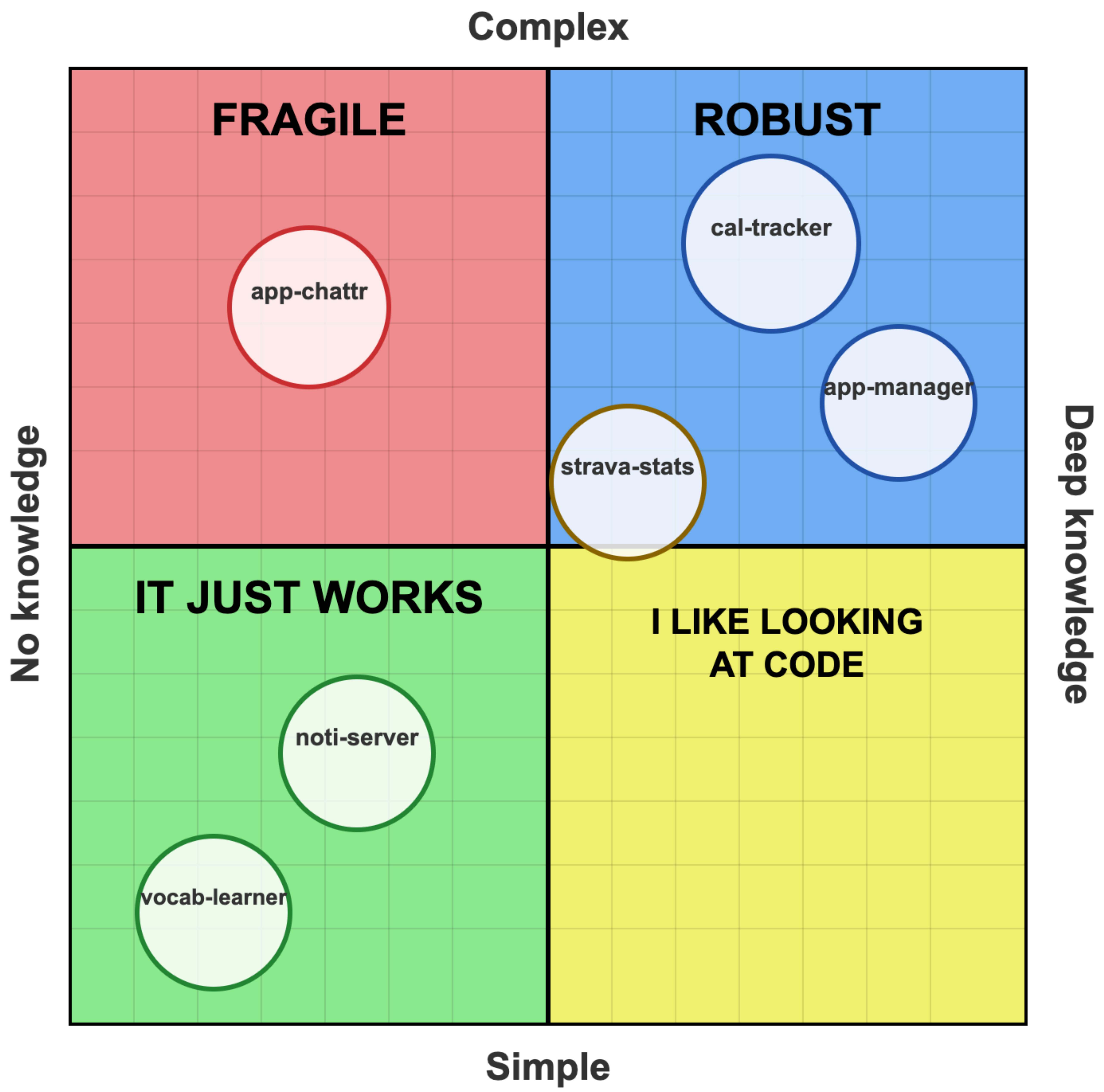
- Ease of planning features
- Waiting on agents anyway
- Every stupid little idea I had in my head could be deployed

Internet / Tailscale
Raspberry Pi 5
app-manager :1200 Gateway + Auth + Dashboard
Libraries (app-manager)
actuator-client starter
Managed Apps
cal-tracker :1337
strava-stats :1342
noti-server :1341
app-transcribe :1344
guitar-practice :1343
app-chattr :1340
vocab-learner :1338
todo-app :1339
Libraries (apps)
strava starter
treatwell starter
App Services (systemd)
ntfy :8090
whisper-server :8091
HTTPS
📱 Phone

Some other quirky apps :-)

By the numbers…

- Around 50.000 LOC
  - 10k in the calorie-tracker
- 8 apps
- 3 libraries
- 223 Claude Code conversations
- Estimated time spent over 2 months: 20-30 hours


Complex
FRAGILE
ROBUST
app-chattr
cal-tracker
app-manager
strava-stats
IT JUST WORKS
I LIKE LOOKING AT CODE
noti-server
vocab-learner
No knowledge
Deep knowledge
Simple

“AI agents can handle long-running coding tasks mostly unsupervised”

“No need to pay for apps anymore, just roll out your own *bespoke* ones”

“Handwriting code is a thing of the past”

Other take-aways

- I **barely** used my IDE
  - AIR by jetbrains
  - conductor
  - cursor
- Cognitive debt
  - ownership-by-refactoring
- Context-switching fatigue
- Burn-out

Things I didn’t try…

- Spec-driven development
    - Marketed as a silver bullet
    - Prefer nonchalant way of working
- Multiple models (ex: Claude for planning, codex for execution)
- Role based agents (‘senior dev’ / ‘architect’ / ‘UX expert’ / …)
- Ralph loop

Thank you!