

€2,234

spent on AI tools from January to October.

I've never spent this much on tools, *and no tool has ever done this much for my work.*

9,341

commits across code,
research and analysis

40+

new repositories for
products and
prototypes

53

client workstreams in a
21-week window, 28
closed

Spend comes from my bank statements (Claude, OpenAI, Google, others). Output comes from GitHub and my client knowledge base.

Anthropic's IPO filing puts my AI spending in context.

Average Claude Enterprise user without Claude Code

 **< \$75 / month**

Top 5% of Claude Enterprise users with Claude Code

 **about 20× (≈ \$1,500)**

€223

my monthly average, Jan—Oct 2026, all vendors

€406

my spend in October 2026

I use several models together and check their results against the sources myself. For me, that combination works better than any single model.

The spending alone doesn't prove anything.

The next slides show what I used it for.

Source: Anthropic PBC, Form S-1, Q3 2026. The filing reports the 20× ratio; \$1,500 is my illustration at \$75. My figures are in euros and cover all AI tools.

As Fractional CPTO, I work in every lane.

Leadership is still the core of the role. With AI, I do much of the first discovery, analysis and delivery myself before the wider team takes over.

Client workstreams by type, 53 in total, clients confidential

Lead & strategy



BEFORE

The main job

2026

Priorities, stakeholders, team setup, workshops and launch playbooks

Discovery



BEFORE

Commissioned, then waited for

2026

Research, surveys, fake-door tests and competitor teardowns

Analysis



BEFORE

Requested from the data team

2026

Dashboards, simulations and models I build myself, cross-checked with other models

Delivery



BEFORE

Tickets for engineers

2026

Prototypes, tools and tests I build myself, and the team takes over what should keep running

Why do it myself? Teams learn this way of working fastest when they see it done on their own problems. When I do the first research or prototype myself, less gets lost in handoffs, and the team has something concrete within days to take over.

With AI, technical depth matters more than before.

AI produces work fast. Whether that work is right depends on someone in the team who knows what to ask and where things break.

DISCOVERY

Knowing what to ask

Which questions a prototype or a test has to answer, and when a model's answer isn't trustworthy yet.

ANALYSIS

Knowing which numbers to doubt

Reading the logic behind a number, spotting what's missing or wrongly included, and checking the result against the source.

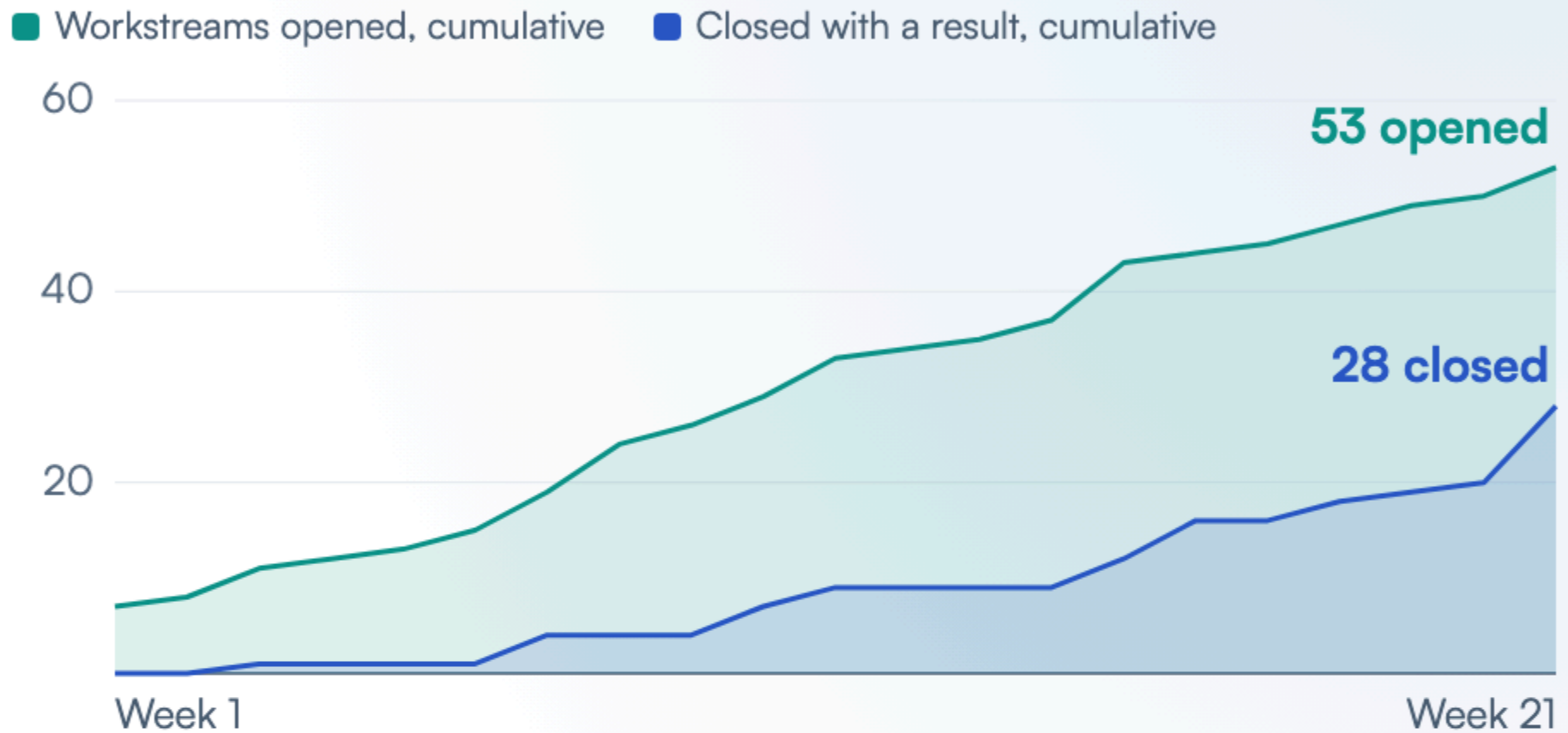
DELIVERY

Knowing which guardrails it needs

Tests, access rights, data protection and a review before anything ships. AI writes the code, but it doesn't set these limits by itself.

In my case, my engineering background tells me what to ask in discovery, which numbers to doubt in analysis and which guardrails delivery needs. **For any team**, someone with that depth has to look at what the AI produces.

Over a 21-week window I opened 53 workstreams and closed 28 of them with a result.



21

weeks

≈ 2.5

workstreams opened
per week

≈ 1.3

closed with a result per
week

I count a workstream as **closed** when a deliverable is finished or a decision is made, including a decision to stop. Workstreams vary in size, so the counts show pace rather than value.

Counts only. Clients are confidential.

Much of the value came from better decisions.

Stop early

An idea that looked promising was stopped before development began, because the data didn't support it.

€ **No development budget went into it.**

Test before you scale

A change was tested with real users first and kept at its current scope.

€ **Bigger investments wait until the numbers support them.**

Build in-house

Something that would have been an external project was built in-house on existing tools.

€ **No external project was needed.**

Use what you already have

What the client already owns is being turned into a new source of revenue.

€ **No new product build is needed for it.**

How we decide: a prototype takes days and helps decide whether to build, buy or partner. The longer-term decision also has to cover support, contracts and staffing.

Client work in 2026. Clients and figures are confidential.

How discovery and analysis work now.

Competitor teardowns

When a new competitor shows up, I have a teardown of its public material, pricing and UX within a few days.

Several models do the research, and I test the product myself.

Data & analysis

KPI dashboards are built as code and refresh daily, so this reporting no longer needs manual spreadsheet exports.

AI writes the code, and running it needs no AI.

Surveys

I can prepare the questionnaire, analysis and deck in the same week, once the responses are in.

Every number can be traced back to the raw data.

Hypotheses on real data

I use historical data to check whether what the client already has could support a new idea.

It also shows what still needs testing.

Prototypes for dogfooding

The team tries clickable prototypes with real data before a decision is made.

They take days to build.

Fast turnaround

A change was tested with real users, the results fit on one page, and a decision followed.

It went from idea to decision within one quarter.

Much of it needs AI only while it is being built.

Built with AI, runs without

Scrapers and data pipelines

Automated end-to-end tests

Document and PDF generation

Dashboards and data cleanup

Internal tools and small apps

These tools make no model calls at runtime, so there is no runtime model bill. Your team maintains them like any other code, and so far AI-assisted changes have kept that effort low.

Runs on AI

Q&A over a document corpus

Agents that score and rank options

Knowledge bases and daily summaries

Personal knowledge assistants

This is worth it when the output itself is text, like answers, summaries or rankings. It needs checks against clear criteria and a person who reviews the output.

One thing to try next week: for every AI idea in your team, ask whether it needs AI to run or only to build.

Pet projects I built next to client work.

In October I updated nine of them from earlier in the year.

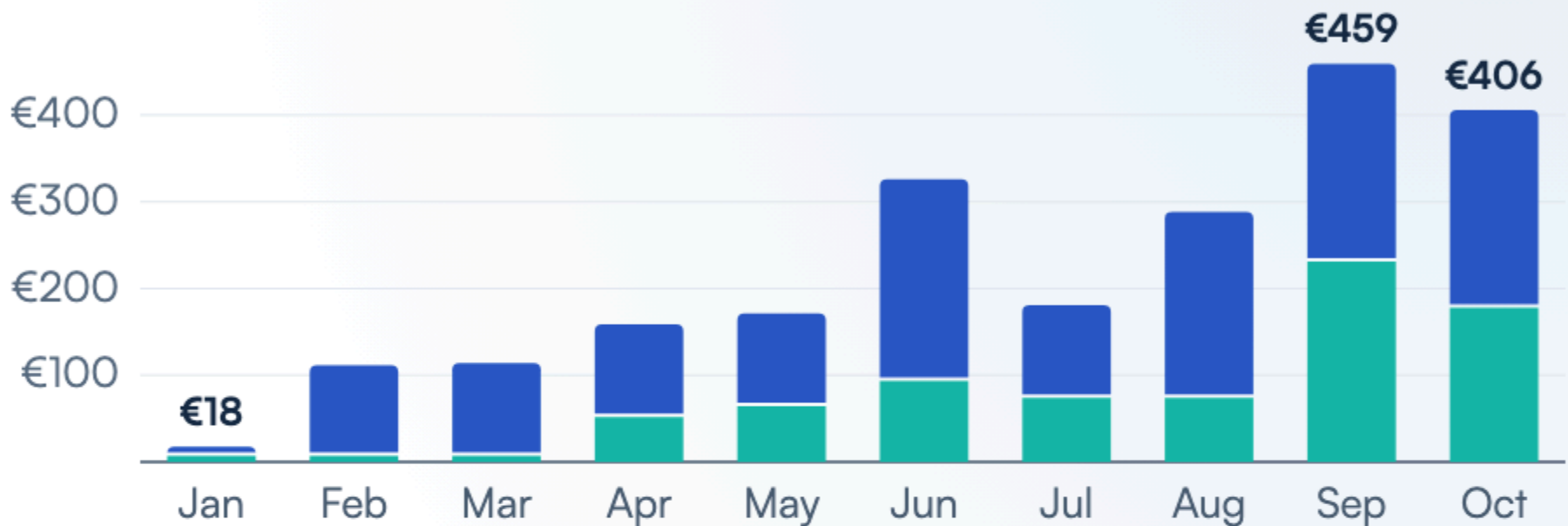
- Q1**  **Jobs mit Impact**
Nonprofit job board for DACH, live, with its own MCP server
jobsmitimpact.de
- Q1**  **Website redesigns**
franzheinfling.com, bitpurpose.de and two more sites, design to deploy
franzheinfling.com • bitpurpose.de
- Apr**  **Agentic-Coop-DB**
Open source: shared Postgres for AI agents, with auth and row-level security
github.com/fheinfling/Agentic-Coop-DB
- May**  **Coreplot**
Local-first AI wiki that writes daily briefings from my sources
- Jun**  **AI Guest List**
Open source macOS app, ships via Homebrew
github.com/fheinfling/ai-guest-list
- Aug**  **Vade**
Local-first knowledge assistant for macOS
- Sep**  **Berlin Atlas**
Berlin by area: crime, elections and residents, from official statistics
- Oct**  **Control + Sempre**
Control center for parallel agent sessions; a menu bar time tracker
github.com/fheinfling/sempre

This selection comes from 40+ new repositories, with links to the public ones.

AI spend and activity, month by month.

AI spend per month

■ Claude ■ OpenAI, Google, others



Commits per month: code, research and analysis



Split across the year, 9,341 commits



Commits measure activity rather than value. The slides before show what the work was.