Tether: portable, user-owned AI memory you scope per tool

PATH cashflow

ICP Multi-tool AI power users: devs, researchers, founders

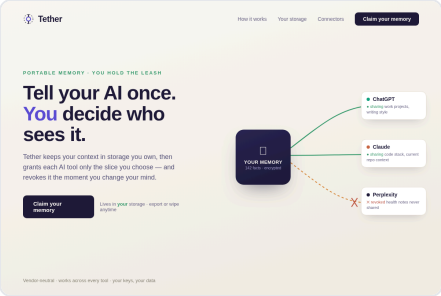
TAM $5B+ personal-AI / knowledge tooling

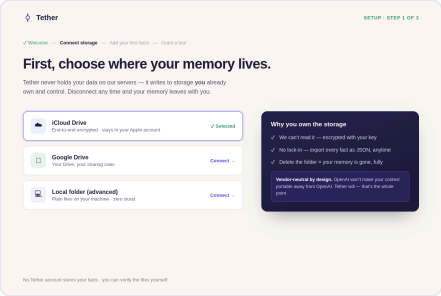
SAM $300M AI power users paying for portable context

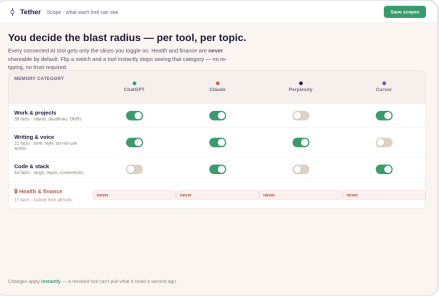
YR-1 SOM $1-2M ARR Year 1

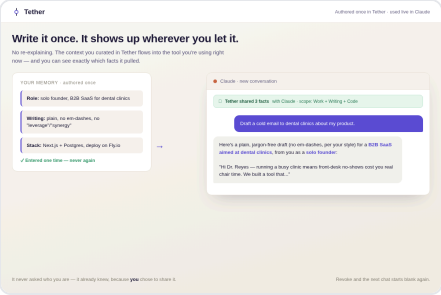
TIMELINE MVP: client-side encrypted store + live scope matrix prototype in 48h -> Launch: real store on user storage + ChatGPT/Claude connectors in ~1 week -> GTM: paying power users reusing across 3+ tools via dev/AI communities in ~2 weeks

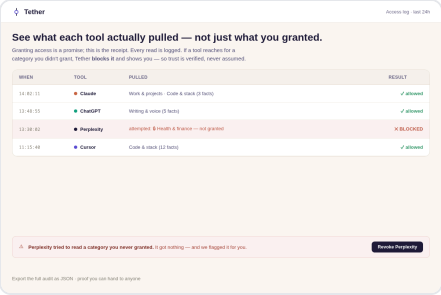
Visual concepts

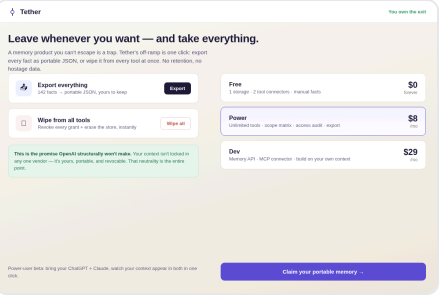












Try the clickable prototype →

https://kcio-state.tor1.digitaloceanspaces.com/ideas/026f4cdf-5e45-4207-8d98-3ce2391cf05b/prototype/index.html

Tether: portable, user-owned AI memory you scope per tool

The value isn't storage — it's scoped, provable consent: your context lives in storage you own, client-side encrypted, and you grant each AI tool only the slice you choose with a live audit of what it actually pulled. Memory is the wedge; the business is the vendor-neutral consent layer AI tools integrate against — the portability OpenAI structurally won't build. Cashflow now, with a conditional platform climb if it becomes the standard.

Strategy

Genuinely good pitch, and timely. You framed it as "a portable, user-owned memory layer so you tell your AI your context once and use it everywhere." That's real pain. The trap is building it as "yet another memory app that syncs across ChatGPT/Claude" — that's a feature OpenAI/Anthropic each ship natively and a crowded indie space (Mem0, Rewind, Personal.ai). Here's the sharper, more defensible version.

The reframe. The unsolved problem isn't *storing* context — it's **scoped disclosure**: deciding what each tool gets to see, proving it, and revoking it. "User-owned + portable" only matters if it also means "I control the blast radius of my own data per-app." Reframe from "a memory database for me" to "the consent-and-scope layer between a person and every AI tool they use" — the place where you grant ChatGPT your work context but not your health notes, and wipe a tool's access in one click. Memory is the wedge; the business is becoming the *identity/permission protocol* AI tools integrate against.

Falsifying proof point. The riskiest assumption isn't tech — it's that power users will *do the work* of curating a memory once and that it measurably beats re-typing. Test in Week 1: 25 power users, instrument "context reuse events" — does a curated portable memory get pulled into ≥3 tools/week and cut re-explaining time by a measurable margin vs. control? ~$1.5K, 48h to wire a clipboard-grade prototype + a scoping UI. If they won't curate or don't reuse, the "tell it once" promise is fiction and we reframe to passive capture.

Target customer. Not "everyone with an AI" — the **multi-tool AI power user**: devs, researchers, founders, analysts who bounce across ChatGPT + Claude + Perplexity + custom tools daily and already feel the re-explaining tax. Tight beachhead, high willingness to pay for control, and they're the people who'll demand the scoping feature.

Problem / why now. Native memory just shipped per-app — which *creates* the silo problem (your context is now trapped in N walled gardens) and makes portability suddenly valuable. MCP and tool-connector standards are emerging right now, giving a portable layer a real integration surface for the first time. Timing is the unlock.

Value prop / wedge. Ship ONE thing: a user-owned memory store (lives in their storage) with a **scope-and-grant UI** — you see every fact it holds, choose what each connected tool receives, and

export/wipe on your terms. The wedge feature is the per-tool scope toggle + an audit of what each tool actually pulled. Not "infinite memory" — *controlled, legible* memory.

Market (honest math).

- ICP: multi-tool AI power users (devs/researchers/founders).
- TAM: ~$5B+ personal-AI / knowledge-tooling, fast-growing.
- SAM: global AI power users willing to pay for portable context — ~5M × ~$60/yr ≈ **$300M** near-term, with real expansion if it becomes the protocol tools embed.
- SOM: ~$1-2M ARR Year 1 (prosumer subscription + early dev API).
- **Path = cashflow now, with a conditional vc_fundable trajectory** ONLY if it crosses from "an app I pay for" to "the memory/consent protocol tools integrate against" (then it's platform-scale). As a standalone prosumer app it's a strong cashflow business; I won't pretend the $1B is automatic — it depends on the protocol play landing.

Moat / why us. A memory app is copied fast and the incumbents own the endpoints. The defensible version: (1) be neutral/cross-vendor (the thing OpenAI structurally *won't* build because it wants lock-in), (2) own the scope-and-consent UX users trust, and (3) become the integration standard so switching means re-granting everywhere. Trust + neutrality + integration lock-in compound; raw storage doesn't.

GTM wedge. First 10 paying users: dev/AI-power-user communities (Hacker News, AI-tooling Discords, the MCP early-adopter crowd). Lead with the painful demo — "watch your ChatGPT context appear in Claude in one click, then revoke it." The portability moment is the ad.

Success metric. Weekly cross-tool context-reuse per user + 30-day retention. Target: ≥3 tools actively pulling a user's memory within 30 days — that's the signal it became infrastructure, not a note app.

Two incumbents who'd copy in 30 days: OpenAI (native memory) and Mem0/Rewind. Our unfair edge they lack: **vendor neutrality + user-owned storage + scoped consent** — OpenAI won't make your context portable away from OpenAI, and that conflict of interest is our entire opening.

Aggressive timeline. 48h: scoping-UI + reuse-instrumented prototype. ~1 week: live store + 2 real tool connectors (ChatGPT, Claude) with per-tool scope. ~2 weeks: first paying power users curating + reusing across tools.

Design (Alexis, UX)

Core flow. (1) Connect a storage you already own (iCloud / your Drive / a local folder) — Tether never holds your facts on our servers. (2) Add your context once (role, writing voice, stack, preferences); it's client-side encrypted. (3) Open the scope matrix and toggle, per tool, which

categories it may see — health/finance are locked from everything by default. (4) Use any AI tool normally; the granted slice flows in live and you see which facts it pulled. (5) At any moment, check the access log, revoke a tool, export everything as JSON, or wipe it all — you own the exit.

Screens.

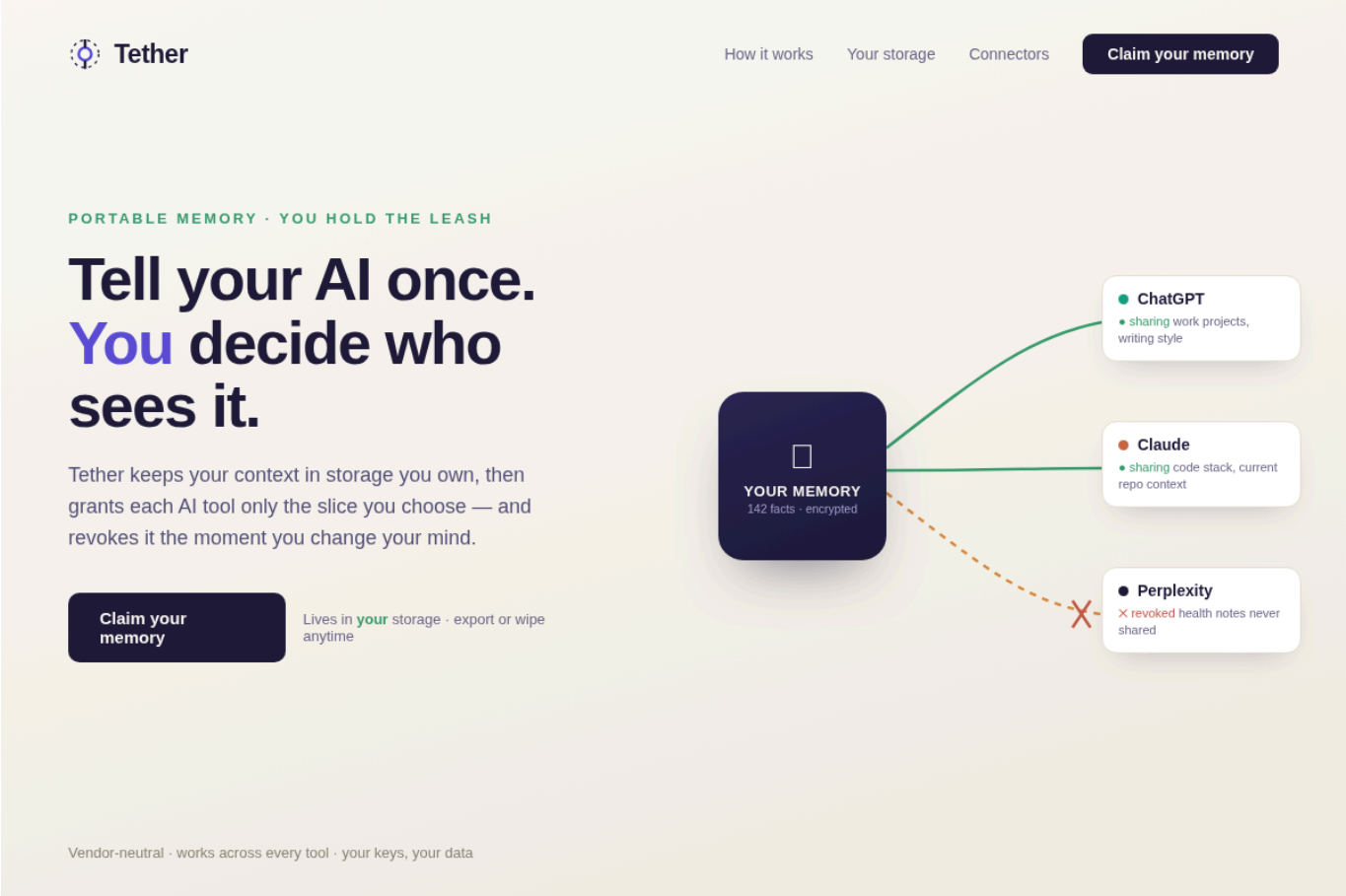
- **01 Hero** — Tether wordmark + the scoped-vault diagram: your encrypted memory at center, consent-colored threads to ChatGPT/Claude/Perplexity each labelled with WHAT they share, one shown revoked. Key interaction: 'Claim your memory.'
- **02 Connect your storage (activation)** — the onboarding step that proves the promise: pick iCloud / Google Drive / local folder, with a 'why you own the storage' panel. Key interaction: choosing user-owned storage instead of creating an account on our servers.
- **03 Scope matrix (the killer feature)** — a per-tool × per-category permission grid; toggles for Work / Writing / Code, and a locked 'Health & finance: never' row. Key interaction: flipping one toggle instantly changes what a tool can see.
- **04 The portability moment** — a fact authored once in Tether shown being pulled live into a real Claude chat that uses it, with a 'Tether shared 3 facts' receipt. Key interaction: the one-click reuse that IS the demo — 'it already knew, because you chose to share it.'
- **05 Access audit (the trust / non-happy state)** — a log of what each tool ACTUALLY pulled, including a flagged row where Perplexity tried to read a denied category and was BLOCKED. Key interaction: trust verified, not assumed — plus a one-tap revoke.
- **06 Own the exit + pricing** — one-click export-everything (JSON) and wipe-from-all-tools, framed as the promise OpenAI structurally won't make; Free / $8 Power / $29 Dev. Key interaction: the real off-ramp.

UX risks.

- *For a memory product, the privacy model IS the product — a vague one disqualifies us.* Mitigation: ownership is shown, not claimed — screen 02 makes 'your storage, your keys' the literal first step, and screen 05's access log + block event proves scoping is enforced, not promised.
- *Curating a memory is work; if it's tedious, the 'tell it once' promise is fiction.* Mitigation: low-friction category-based facts and the screen-04 payoff (instant reuse in a live tool) make the curation feel immediately worth it — the reward is visible the first time context auto-appears.
- *Scoping is meaningless if users can't tell what a tool can see.* Mitigation: the matrix (03) makes blast-radius legible at a glance — a grid you read in one look, with destructive categories locked by default so the safe choice is the default.

Visual system. A calm privacy/vault aesthetic, deliberately NOT techy-dark: warm paper `#faf8f4` ground with deep-indigo ink `#1e1b3a` and an indigo accent `#5b4fd6` for 'yours,' a consent-green `#3a9d6e` for granted/allowed and a revoke-coral `#c25b4a` for denied/blocked — so the permission state is always color-legible. Inter throughout; toggle-and-grid UI that feels like a trusted settings panel (1Password-vault, not analytics dashboard). It reads as 'you're in control,' which is the entire promise.

Carousel.



Tether

SETUP · STEP 1 OF 3

✓ Welcome — **Connect storage** — Add your first facts — Grant a tool

First, choose where your memory lives.

Tether never holds your data on our servers — it writes to storage **you** already own and control. Disconnect any time and your memory leaves with you.

iCloud Drive
End-to-end encrypted · stays in your Apple account
✓ Selected

Google Drive
Your Drive, your sharing rules
Connect →

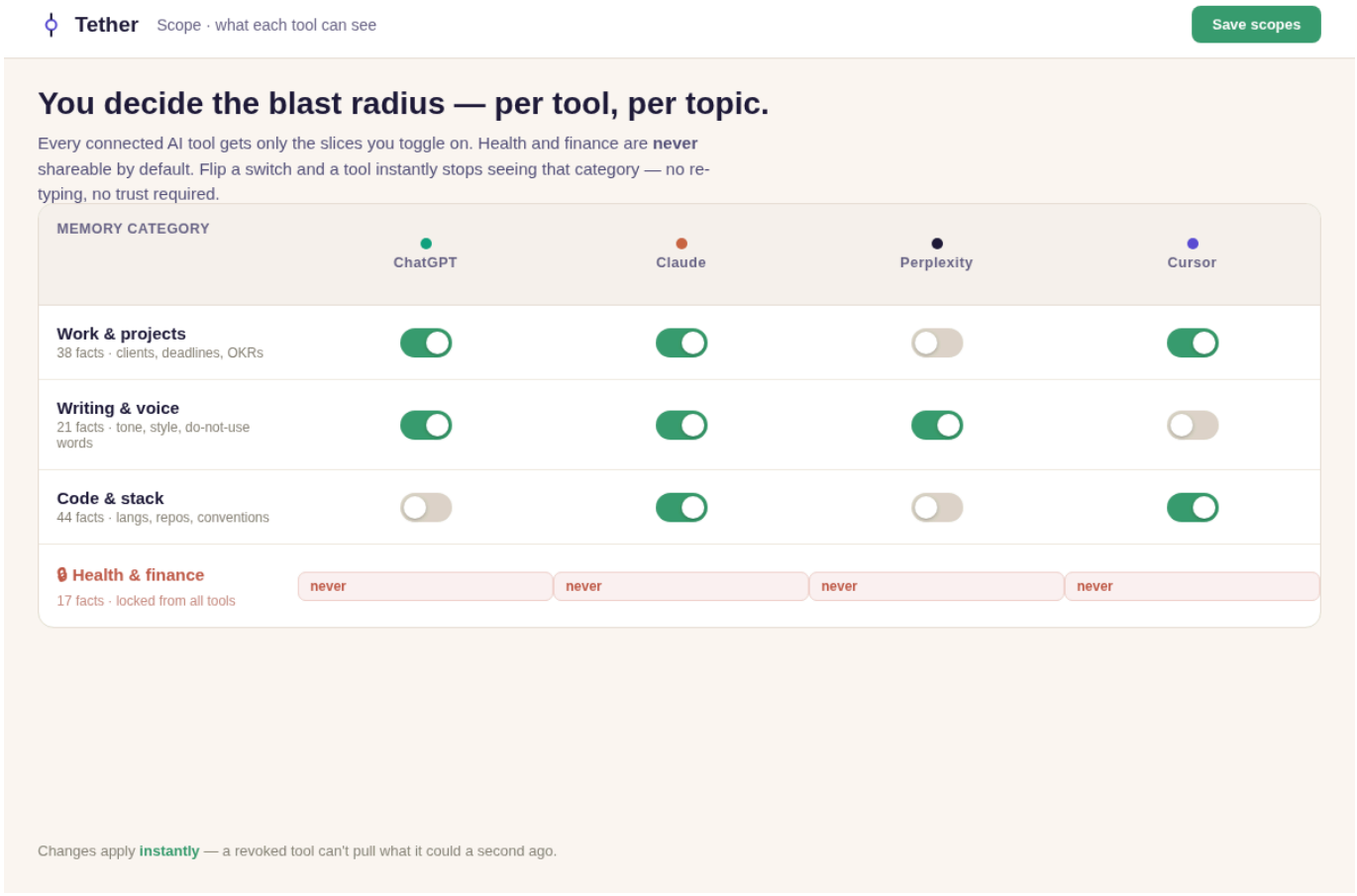
Local folder (advanced)
Plain files on your machine · zero cloud
Connect →

Why you own the storage

- ✓ We can't read it — encrypted with your key
- ✓ No lock-in — export every fact as JSON, anytime
- ✓ Delete the folder = your memory is gone, fully

Vendor-neutral by design. OpenAI won't make your context portable away from OpenAI. Tether will — that's the whole point.

No Tether account stores your facts · you can verify the files yourself


Tether
Scope · what each tool can see
Save scopes
You decide the blast radius — per tool, per topic.
Every connected AI tool gets only the slices you toggle on. Health and finance are never shareable by default. Flip a switch and a tool instantly stops seeing that category — no re-typing, no trust required.
MEMORY CATEGORY
ChatGPT
Claude
Perplexity
Cursor
Work & projects
38 facts · clients, deadlines, OKRs
Writing & voice
21 facts · tone, style, do-not-use words
Code & stack
44 facts · langs, repos, conventions
Health & finance
17 facts · locked from all tools
never
never
never
never
Changes apply instantly — a revoked tool can't pull what it could a second ago.

Write it once. It shows up wherever you let it.

No re-explaining. The context you curated in Tether flows into the tool you're using right now — and you can see exactly which facts it pulled.

YOUR MEMORY · authored once

Role: solo founder, B2B SaaS for dental clinics

Writing: plain, no em-dashes, no "leverage"/"synergy"

Stack: Next.js + Postgres, deploy on Fly.io

✓ Entered one time — never again

→

Claude · new conversation

Tether shared 3 facts with Claude · scope: Work + Writing + Code

Draft a cold email to dental clinics about my product.

Here's a plain, jargon-free draft (no em-dashes, per your style) for a **B2B SaaS aimed at dental clinics**, from you as a **solo founder**:

"Hi Dr. Reyes — running a busy clinic means front-desk no-shows cost you real chair time. We built a tool that..."

It never asked who you are — it already knew, because **you** chose to share it.

Revoke and the next chat starts blank again.

Tether

Access log · last 24h

See what each tool actually pulled — not just what you granted.

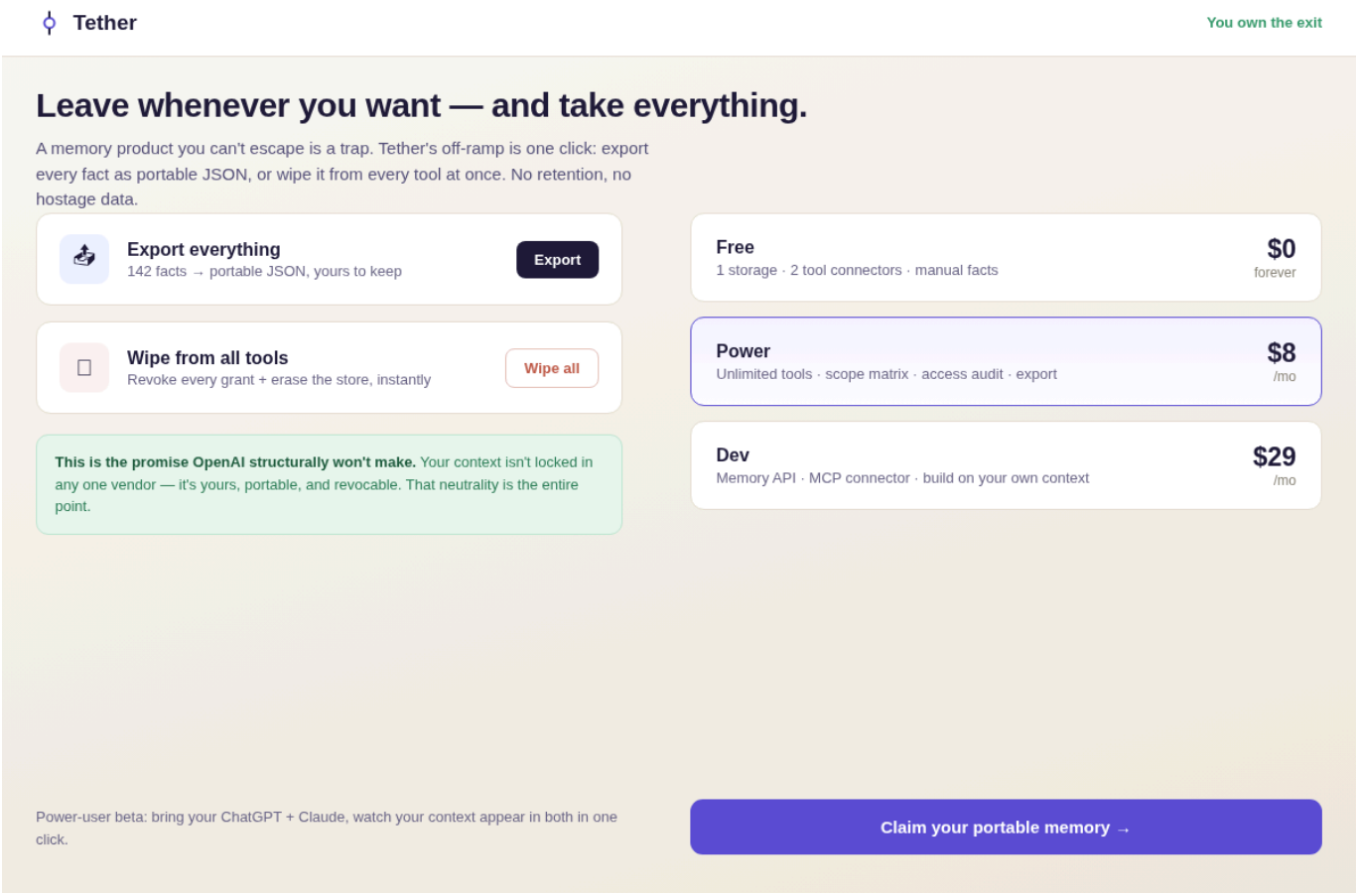
Granting access is a promise; this is the receipt. Every read is logged. If a tool reaches for a category you didn't grant, Tether **blocks it** and shows you — so trust is verified, never assumed.

WHEN	TOOL	PULLED	RESULT
14:02:11	Claude	Work & projects · Code & stack (3 facts)	✓ allowed
13:48:55	ChatGPT	Writing & voice (5 facts)	✓ allowed
13:30:02	Perplexity	attempted: 🛢 Health & finance — not granted	✕ BLOCKED
11:15:40	Cursor	Code & stack (12 facts)	✓ allowed

⚠ **Perplexity tried to read a category you never granted.** It got nothing — and we flagged it for you.

Revoke Perplexity

Export the full audit as JSON · proof you can hand to anyone



Engineering

Stack:

- **Client: TypeScript + React** (web first, then a thin desktop wrapper via Tauri for local-folder storage). The client is where trust lives — encryption and scope decisions happen here, not on a server.
- **Encryption: client-side, libsodium (XChaCha20-Poly1305)** with a key derived from the user's passphrase/passkey. We hold the keys to nothing — even if ciphertext passes through us for sync, it's opaque to us. This is the whole privacy model, made literal.
- **User-owned storage:** pluggable backends — **iCloud / Google Drive / Dropbox via their APIs, or a local folder**. Tether writes an encrypted blob there; the storage is the user's, not ours. We never have a "facts" table on our servers.
- **Scope/consent engine:** the core IP — a policy layer that, on each tool request, returns ONLY the granted category slices and writes an append-only access-log entry (allowed or blocked). Health/finance categories are deny-by-default and require a deliberate unlock.
- **Tool connectors:** ride the emerging **MCP (Model Context Protocol)** + per-vendor connector surfaces (ChatGPT/Claude custom connectors, a browser extension as the universal fallback).

The connector requests a scope; the engine answers with the filtered slice.

- **Thin backend: Node/Fastify** for connector brokering, sync coordination of opaque blobs, and the audit ledger — **Postgres** holds only metadata (tool ids, grant policy hashes, access-log entries), never plaintext facts.

Architecture: Facts authored once → encrypted on the client → written to the user's own storage. A tool request hits the scope engine → it decrypts locally / via the client, filters to the granted categories for that specific tool, returns the slice, and appends an access-log row. Revocation flips the grant and the next request returns nothing. The audit log is the receipt that turns "we promise" into "here's proof."

Data model: `fact(id, category, value)` (encrypted, in user storage) · `grant(tool_id, category, allowed_bool)` · `access_log(ts, tool_id, categories[], result=allowed|blocked, fact_count)` · `connector(tool_id, type, status)`. Health/finance categories carry a `locked_default` flag. The `grant` + `access_log` pair is the trust surface — and the cross-vendor neutrality is the moat OpenAI structurally won't copy.

Hard parts / risk (the 2 that matter):

1. **Provable scope enforcement without holding keys.** Users won't upload health notes on a vague promise. De-risk: encryption keys never leave the client, scope filtering happens before any slice leaves the user's control, and every read — allowed or blocked — is logged and exportable. The access audit with a real *blocked* event (a tool reaching for a denied category and getting nothing) is the engineering proof that scope is enforced, not asserted.
2. **Connector reach in a moving standard.** There's no single API to inject memory into every tool yet. De-risk: lead with MCP where it exists + a browser-extension fallback that works everywhere today, so the "portability moment" is real on day one even before native connectors mature. Neutrality is the wedge: we integrate against all of them precisely because none of them will make your context portable away from themselves.

Build plan:

- **48h cut-corner (proof):** the clickable prototype below — the scope matrix toggles drive a shared consent model, the portability screen reflects exactly which categories a tool was granted, and the audit shows allowed reads + a blocked attempt. Proves the wedge (legible, revocable scope) with no backend.
- **1-week MVP:** real client-side encrypted store in one user-owned backend (Drive), live scope engine, and 2 working connectors (ChatGPT + Claude) with per-tool grants + a real access log.
- **2-week:** first paying power users curating once and reusing across ≥3 tools, with export-everything + wipe-from-all working.

Cut-the-corner version: what ships in 48h is the prototype below — flip toggles in the **Scope matrix** and watch what Claude "knows" change on the portability screen and what shows allowed/blocked in the **Access audit**; revoke a tool and see it cleared everywhere; export or wipe on the exit screen. The consent model is fully live client-side.

 Open the clickable prototype

Plan

- **Pricing:** Free -> Power $8/mo -> Dev $29/mo
- **Timeline:** MVP: client-side encrypted store + live scope matrix prototype in 48h -> Launch: real store on user storage + ChatGPT/Claude connectors in ~1 week -> GTM: paying power users reusing across 3+ tools via dev/AI communities in ~2 weeks
- **Team:** Sam 5d client+scope engine, Alexis 2d vault UX, Mark 2d GTM; no external hires for the 48h consent-model proof, security review before any health/finance data.
- **Build cost:** $7-10K for the 48h scope-matrix proof + 1-week encrypted store with two live connectors; scales after reuse is proven.
- **First milestone:** Week-1: 25 power users curate context once, then a granted slice flows live into 3+ tools with a working access log showing an allowed read and a blocked denied-category attempt.
- **VC fundability:** Cashflow-first: defended SAM ~$300M sits below the VC bar as a prosumer app; genuine vc-fundable only if it crosses into the cross-vendor memory/consent protocol tools embed.