402Guard

402Guard Pitch Deck

Guardrails for x402 payments and subscriptions on Avalanche

Dec 12, 2025

Avalanche x402 Hack2Build | Payments: Tooling & Infrastructure

Target User

x402 makes API payments simple. But teams shipping agents and paid endpoints still need controls: budgets, rate limits, plan tiers, and clear spend attribution. 402Guard is built for builders who want x402 without “runaway spend” risk.

Agent Builders

AI agent teams and backend engineers who let software pay for data, tools, and compute. Need safe defaults and audit trails.

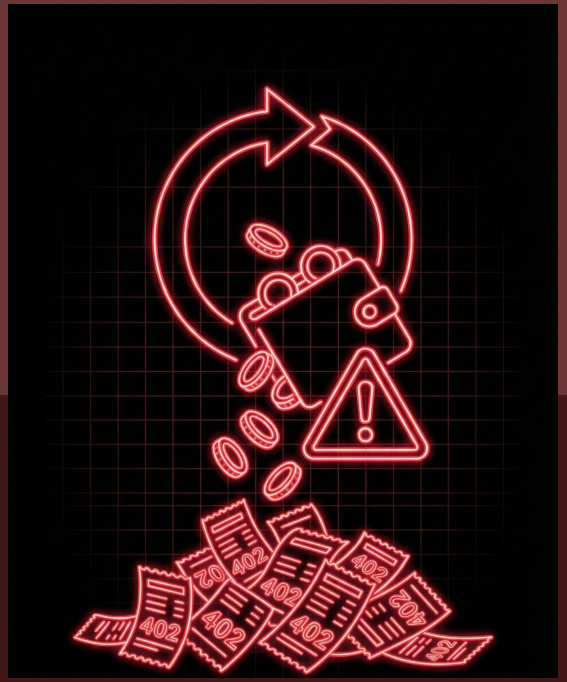
API Providers / Marketplaces

Teams selling paywalled APIs that want “Netflix-style” plans, caps, and invoices without building billing infra from scratch.

The problem

x402 tells you how to pay. It does not tell you when you should not pay.

- **Runaway agents**: valid 402 payments can drain wallets fast.
- **Missing SaaS primitives**: plans, tiers, quotas, per-seat controls are not built-in.
- **Opaque spend**: “which agent spent how much on which service” is hard to answer.



Our solution

402Guard is a safety + finance-ops layer for x402. Wrap your existing Axios/fetch client, define policies in USD, and only authorize payments that pass guardrails. Track usage by agent and subscription, then generate invoices.

How it works

Guarded Requests

Intercepts HTTP 402, checks caps, then pays and retries automatically.

- Prevents accidental overspend before signing
- Same interface for 200-only and 402 flows

Policies in USD

Budgets + rate limits per agent, service, and endpoint.

- Daily/monthly caps, per-service caps
- Allow/Block decision is explicit and logged

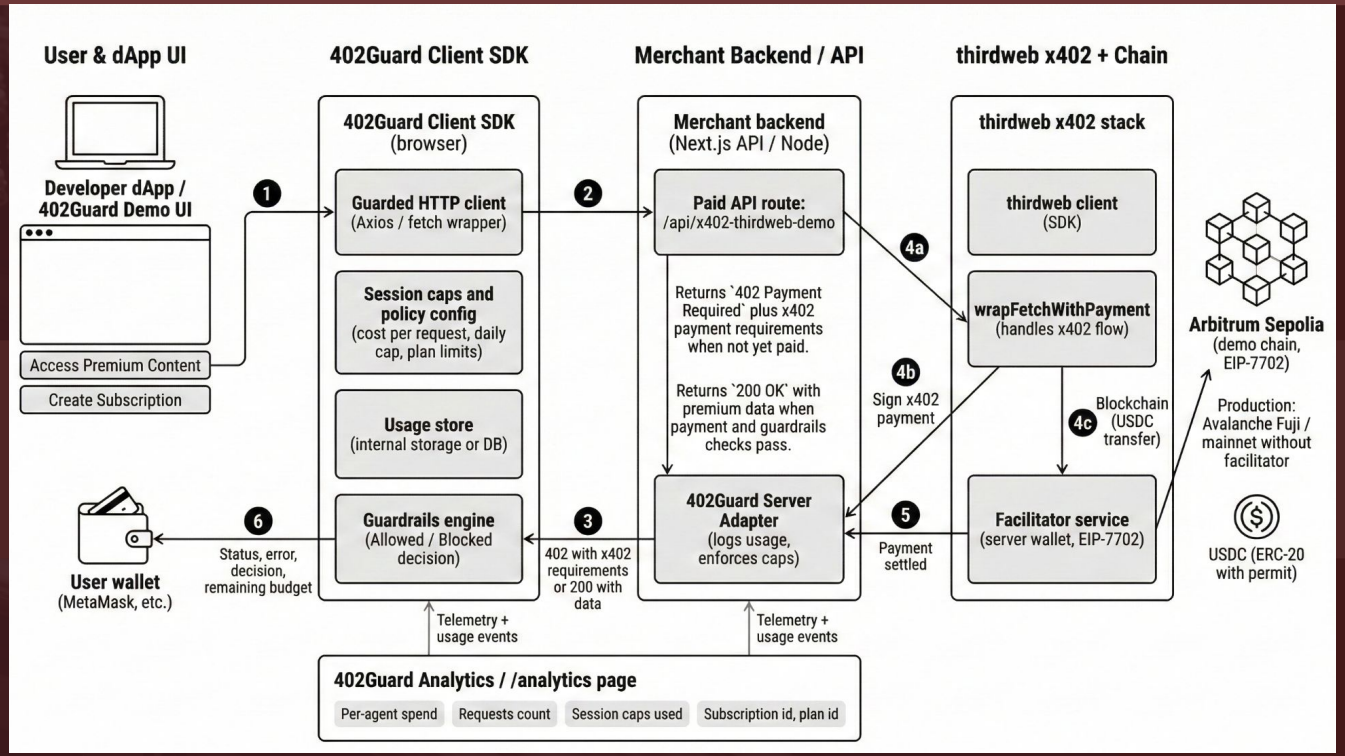
Subscriptions + Invoices

On-chain subscription validity + local enforcement + downloadable invoices

- Plans map to caps (Starter/Pro)
- Usage becomes finance-ready records

Architecture Overview

Hybrid design: on-chain subscription truth on Avalanche, off-chain policy engine for fast, granular control.



Client SDK (@402guard/client)
- Guarded axios/fetch wrapper
- Policy engine + UsageStore hooks

On-chain Subscriptions
- Plans + active subscribers
- Avalanche Fuji source of truth

Analytics + Invoicing
- Spend by agent/subscription/API
- Export invoice + audit trail

Roadmap Milestones

We shipped the core SDK + demos.

Next is making this production-grade for real agent stacks: server middleware, durable storage, alerts,
Avalanche-native settlement paths
And an actual npm package.

1.MVP

Guarded client + policies

Subscriptions demo + invoice export

2. + Features

Server middleware for paid routes

Subscription checks at the API edge

3. Prod

Postgres/Redis UsageStore adapters

Webhooks/alerts for spend anomalies

4. Ready

Team seats + delegated budgets

Multi-merchant policy packs + compliance exports

DEMO

Click here

Go-to-market Strategy

Objectives

- Make x402 safe for agents and backend loops
- Become the default “guarded HTTP client” devtool for x402

→ SDK installs (npm) + GitHub stars
→ Weekly active guarded requests
→ teams running subscriptions + invoices

Markets

- AI agent builders (walleted agents, tool callers)
- API providers and x402 merchants
- Infra teams shipping agent runtimes and developer platforms

→ Agent teams
→ API platforms
→ L2/EVM ecosystem builders

Launch

- Hack2Build demo + open-source release (SDK + contracts + docs)
- Recruit 5 pilot teams to integrate in 1 day
- “Guarded x402 starter kit” example repo + short walkthrough video

Channels

- GitHub + npm + docs site (copy-paste quickstart)
- Avalanche + x402 community Discords, X threads, DevRel shoutouts
- Partnerships: agent frameworks, API marketplaces, wallet SDKs

Post-launch plan

- Ship server middleware + durable UsageStore (Postgres/Redis)
- Add anomaly alerts + exportable billing reports
- Publish 3 case studies + integration templates (agents, providers, marketplaces)

Meet the team

Pradyumna Kaushal
Founder | Full-stack + Protocol Engineer

GPT 5.2 Thinking
Creative Director

Gemini Nano Banana Pro
Visuals Director

I built 402Guard end to end: TypeScript SDK, policy engine, usage tracking, and the Avalanche subscription contract, plus the live demo. I focus on the real failure mode in agent payments: valid spend that should never be authorized. My goal is to make x402 production-safe with defaults teams can trust, and analytics finance teams can actually use.

I'm well positioned to build 402Guard because I've worked at the intersection of wallets, agents, and verifiable access control. I built wallet and escrow primitives (multisig, MPC, HD derivation) and shipped privacy-preserving, blockchain-anchored architectures with ZK proofs and access tokens. I've also built an agentic payment pipeline for crypto transactions, so I've seen how automation can turn into unintended spend without guardrails.

pradyumnakaushal.in

THANK YOU!