DANIEL ESTEVEZ

AI Systems Engineer

estevez.dani@gmail.com | linkedin.com/in/estevezdani | github.com/daniboomerang | attalabs.dev | vinaya.attalabs.dev
Remote (GMT+7) — EU/Spanish citizen · Flexible overlap with US hours

PROFESSIONAL SUMMARY

AI systems engineer building TypeScript-first agent systems: coding-agent orchestration, sandboxed execution, CI quality gates, automated review, human-in-the-loop control and LLM observability. 18 years in software, four hands-on with AI.

I built Vinaya, a development harness that runs the whole engineering workflow as a pipeline of agents: a planner writes the task, Claude Code or Codex builds it, two independent reviewer agents judge it, and deterministic gates decide what counts. Every act is recorded, so the system can be watched live, reviewed after the fact and measured with evals. I ship product features through it every day.

The hard part was never the models; it was knowing whether to trust what they produced. Agents fail constantly — they stall, misreport and claim work they did not do — so in Vinaya an agent never decides a fact: code computes it, gates enforce it, the log records it, and a human approves what matters.

Deep product craft on top (React since 2016, Next.js since 2020): I build both the agent system and the interface people use to oversee it.

CORE SKILLS

Agent Harness & Governance: Unattended Dev–Review Loops, Agentic Loop Design (termination, escalation), Execution State Machines, Deterministic Gate Design, Verdict Binding & Stale-Approval Prevention, Reviewer Isolation, Isolated Subagent Context, Typed Handoffs, Human-in-the-Loop Rulings, Pre-commit and CI Enforcement, Git Forge Automation (GitHub CLI/API)

Agent Runtimes: Claude Code & Codex headless with structured outputs, MCP Tool Servers, OS Sandboxing (macOS Seatbelt, Linux bubblewrap/seccomp), Subscription-Auth Agent Dispatch, Provider-Neutral Agent Contracts

AI & Agent Systems: LangGraph, Multi-Agent Orchestration, Multi-Vendor LLM Routing, Model Context Protocol (MCP) Server Design & Hosting, Vercel AI SDK, Claude/OpenAI/Gemini/Grok APIs, YAML-Driven Agent Specs, Streaming LLMs

LLM Observability & Evaluation: Typed Append-Only Event Logs, Live Event Delivery (Cloudflare Workers, Durable Objects, WebSocket), Run Analytics (cost per task, reviewer strictness, rounds to convergence), Agent Evals on Harbor (containerised trials, generic and process-specific graders, oracle checks)

AI Product Interfaces: Agent Run & Trace Views, Live Agent-Execution Graphs (React Flow), Streaming Chat UIs, Prompt Composers with Attachments & Interruptible Streaming, Model Pickers over Live Catalogs, Multi-Agent Deliberation Views, Readable Run Logs

Frontend: React (since 2016), Next.js App Router (since 2020), TypeScript, Tailwind CSS, CSS Modules, shadcn/ui, Radix UI, Ariakit, Storybook, Electron, Figma, TanStack, State Management (Zustand, Recoil, Redux, GraphQL/Apollo), Testing (Jest, React Testing Library)

Backend & Infra: Node.js, Bun, Turborepo, Postgres, Drizzle ORM, Neon, Sanity CMS, Upstash Redis, Clerk Auth, Cloudflare, Vercel, CI/CD

Also: Canvas 2D, WebGL (three.js), Motion (Framer Motion), Core Web Vitals, Storybook, Design Systems, Multi-Tenant SaaS UI, Web3 Wallets (Phantom on Solana, EVM), Electron, Technical Leadership (team of four), Code Review, Testing (Jest, React Testing Library)

Spoken Languages: Spanish (Native), English (C1), French (C1)

PROFESSIONAL EXPERIENCE

AI Systems Engineering | Independent | 2026 – Present

Remote — AttaLabs (attalabs.dev): independent applied AI research and engineering practice

- **Vinaya, a governed delivery loop for coding agents (vinaya.attalabs.dev):** I write the spec; agents ship the feature. Each task runs unattended — Claude Code or Codex develops inside an OS sandbox, deterministic gates run locally and in CI, then two isolated reviewer agents (code and security) judge the same frozen input. The loop converges or pauses with a typed reason and asks a human for a ruling.
- **Trust mechanics:** every verdict is bound to the exact commit, objectives version and rulings it judged — any push or rule change voids it. Missing or malformed reviewer output never counts as approval. Evidence freshness, scope and premise checks catch agents claiming work they did not do.
- **Public record:** the dev log of an app built through the loop, generated from its run records: tickets merged, review rounds and problems caught before merge — zerion.attalabs.dev (demo) · zerion.attalabs.dev/dev-stats (results)
- **Run records:** a typed, append-only event log of every governed act (dispatch, checks, reviews, repository writes, tokens, duration), streamed live through a Cloudflare Workers + Durable Objects server, with analytics over it: completion by model, reviewer strictness, cost per task, check determinism.

- **Atta Engine:** my workflow engine: a YAML flow compiled into an immutable plan and run on LangGraph, with checkpoints, typed halt and resume, stop conditions, live events and cancellation; at every step it calls my own drivers — an agent role with exactly its own tools and permissions, or a deterministic check. Vinaya runs on two backends: its standalone driver and Atta Engine.
- **Agent evals on Harbor:** every Vinaya role runs as a Harbor agent in a container trial, graded on outcomes by generic graders plus Vinaya-specific ones, with an oracle check that each case is solvable; the same cases and graders run against both execution backends, on Claude and Codex.
- **Vāda deliberation engine:** multi-agent deliberation on LangGraph with multi-vendor routing (Claude, Gemini, GPT, Grok), cost/quality fallback and per-model cost tracking; a hosted MCP server with envelope-encrypted BYOK and bearer auth.
- **Product surface:** Vinaya Studio (Next.js) over the harness, and Mission Control, a live view I designed over the event log — one row per run, so a dead run never looks paused and unknown never shows as zero. Shared design system: four interchangeable libraries held to one enforced contract, 141 components.

Senior Frontend Architect | Game7 (Summon) | Nov 2023 – Mar 2026

Remote | United States — Web3 gaming platform with on-chain quests, rewards, and multi-chain wallet integration

- Built a prompt-to-landing-page feature — users describe a site and get one — on a three-agent pipeline (site-type classifier, section composer fanning out parallel low-cost model calls, synthesiser producing a page tree) stored per tenant in the CMS and rendered against the component contract
- Designed multi-agent AI content generation with parallel orchestration and model routing matched to task complexity, optimising speed, cost and output quality together
- Built an OpenAPI → component props pipeline so agents consume real data shapes — the architectural precursor to MCP
- Established the team's AI-augmented development workflow — structured specs, agent-assisted code generation with human review gates, reproducible patterns for non-deterministic outputs
- Architected a multi-tenant SaaS platform (React, Next.js, TypeScript, Turborepo, Zustand, Sanity CMS; tested with Jest and React Testing Library) with swappable UI libraries behind type contracts, resolved by build-time tenant generation with no runtime branching
- Delivered dynamic theming with perceptually uniform colour science and real-time multiplayer features (Colyseus, WebSocket, server-authoritative state sync)

Lead Frontend Engineer | Elixir Games | Feb 2021 – Nov 2023

Remote | San Francisco Bay Area — Web3 gaming launcher and dApp platform

- Led a team of four frontend engineers; made code review the centre of how the team worked, so it learned from each other and built shared conventions.
- Owned the releases of the Elixir Launcher for macOS, Windows and Linux.
- Led frontend architecture decisions, code quality ownership, and technical direction across all products
- Built the Elixir Launcher from scratch: cross-platform Web3 game launcher (Electron, Next.js, Recoil, React Query; tested with Jest and React Testing Library) shipped on Windows, Linux, and macOS
- Developed the web launcher and Developer Dashboard (Next.js) for game publishing and management, with analytics integration and SEO improvements
- Led frontend for two blockchain gaming dApps — Monstropoly and No Way Back — with multi-chain integration (Ethereum, Polygon, Arbitrum)
- Integrated the Phantom wallet on Solana alongside EVM wallet support

Earlier Experience

- **Frontend Engineer, Sherpany** (Zurich, remote) | Aug 2020 – Feb 2021 — board management SaaS; TypeScript, React, Redux, accessible UI components
- **Frontend Developer, Conferize** (Copenhagen, remote) | Feb 2016 – Feb 2020 — primary frontend engineer for four years; React, GraphQL (Apollo), real-time features, Jest/Enzyme test suite
- **Full Stack Developer, Freelance** (worldwide, remote) | May 2013 – Jan 2016 — end-to-end web apps for international clients; MEAN stack
- **Software Engineer, Amadeus IT Group** (Sophia Antipolis, France) | Sep 2010 – Mar 2013 — R&D on the core booking engine across UI, business logic and GDS integration
- **Software Engineer, Zennio** (Toledo, Spain) | Oct 2008 – Jan 2010 — EIB/KNX home automation; embedded firmware in C and C++, tooling in J2EE and LabView

EDUCATION

Computer Science Engineering (Ingeniería Informática) — Universidad Politécnica de Madrid (UPM) | 2001 – 2008
Five-year engineering degree — the pre-Bologna Spanish Ingeniero en Informática qualification, equivalent to MSc level.
Erasmus Exchange — University of Thessaly, Greece · Athens Programme — Delft University of Technology, Netherlands