

Sherman Rose

Senior Software Engineer – Backend & Platform

PROFILE

Nearly fifteen years building software, starting with the tools I wrote to do my Linux support job. Today I build backend APIs and infrastructure day to day, reaching into data on one side and AI enablement on the other. .NET for the last two years, Go for nearly three, on PostgreSQL, messaging and Terraform/GCP.

Writing code is close to solved. The unsolved part is designing systems that stay testable, and keeping velocity without losing stability. Code has no value in itself: its value is what it enables, or what people believe it enables. I treat it as the means to that end, and manage it so it never gets in the way.

I work AI-first. Agents do most of the typing; I design the system, the gates they work inside and the checks that prove their output, and I own the calls that can't be undone.

Ask me why, in the era of AI, everything we knew about building software matters more, not less.

HOW I WORK WITH AI

Verify, don't trust. Tests and checks, not vibes.

Humans own the irreversible. Specs, merges, releases.

Specs before code.

Cheapest model that can do the job.

Encode judgement as skills, not tribal knowledge.

One write path, hard boundaries.

WHAT I'M EXPLORING

Hem: a true ADLC, one spec graph and one write path for agents and humans

Ticket-to-PR agents where only a human approves and merges

Multi-agent orchestration across parallel worktrees

Model tiering and the cost of agentic work

Verification gates for agent output

CONTACT

sherman@shermanrose.uk
github.com/hiniko
United Kingdom

EXPERIENCE

Oct 2024 – Present

Senior Software Engineer, Backend & Platform – Catches

Early-stage AI start-up (B2B try-on and media generation). Day job: backend APIs and infrastructure. In practice: helping bring up the platform across the whole stack, from the data warehouse to company-wide AI enablement.

Platform & architecture

- Go-to engineer for **requirements gathering and systems design on cross-team projects**: regularly asked into other teams' design sessions, and designed and helped spec the systems they build on.
- Designed the **media-generation core** of the platform: set its architecture and principles, held them through review, and moved render traffic onto it with a parity suite against the previous renders.
- Built the **asset-pipeline versioning system**: a generalised versioned-asset model and the registration API the content pipeline publishes through. Designed and built bulk render and AI job handling end to end, with live progress.
- Largest contributor to the .NET backend**, author of over half its current code across APIs, services, consumers, data and tests.
- Built a **hermetic black-box test environment** for rapid AI and API correctness testing: it catches breaking API changes before they ship, so the team can react early, deliver faster and move towards continuous deployment. Top author of the release and CI pipeline.
- Also: led an API migration from Go to .NET; layered configuration; audit logging with PII redaction; notifications; entitlements; accounts and tenants.

Infrastructure & shared services

- Built **monitoring as code** in Terraform: 5xx and latency monitors, alert routing and a standard dashboard for every platform environment; redesigned the RabbitMQ queue monitors to cut alert noise.
- Driving cross-team ownership of shared services** (RabbitMQ, monitoring): team-owned queue definitions with alerts routed to the owning team. Provisioned GCP infrastructure (storage, load balancers, DNS, Kubernetes consumers).

Data

- Founded the company data warehouse**: dbt on BigQuery, medallion conventions, CI/CD, docs and a Lightdash semantic layer; a render-job **cost model**; platform-telemetry ingestion (Pub/Sub to BigQuery) with its Terraform. A dedicated data engineer now builds on these foundations.

Admin Portal

- Rebuilt the Admin Portal from first principles** in the shared Next.js monorepo and wrote most of it; other engineers now build on its architecture. Over a hundred routes covering tenant, user and role administration, product and render management, and platform settings.
- A highly reusable **MVVM/SPA-style architecture**: most routes are shared components that work as soon as new models are added, so new areas come up quickly. **Designed AI-skill-first**, so other team members can add their features as fast as possible. Removed its server actions after real-user monitoring showed the latency cost.

Agentic engineering & enablement

- Authored **~30 agent skills** across the backend platform, the Admin Portal and the company plugin repo, so engineers can work quickly in familiar and unfamiliar codebases alike: from API, data and messaging conventions to a spec-gated, eval-tested feature workflow. Includes a skill-authoring standard and a drift check that keeps agent guidance in step with the code.
- Built the company's **Claude plugin platform**: lint-gated, versioned per plugin and published org-wide to Claude Code and Claude Desktop. Now being picked up across the company: the first plugin from another engineer has already shipped.
- Wrote a cost-routing skill that sends agent work to the cheapest model that can do the job.

C#/.NET, ASP.NET Core, EF Core, Aspire, Go, PostgreSQL, RabbitMQ, Redis, Terraform/GCP, dbt/BigQuery, Next.js/React, Datadog, Claude Code.

SIDE PROJECTS

Research on the same thesis. Private repos/demos available on request.

Hem – my take on AI-first product development through a true ADLC (agentic development lifecycle) system, rather than several systems patched together. Specs, tickets, agents, code and verification in one closed loop. Its core is the spec layer as one typed graph, with a single write gate that agents and humans both go through.

autoplane + Plane skills – tickets as the human↔agent contract. Agents take only tickets a human marked Ready; a circuit breaker tells an outage from a red build; three red verifies and it goes back to a human.

Cobble – a game-systems modelling language that unites simulation and statistical-analysis methods to prove out game systems before opening an engine, tuning balance and feel as a process separate from building the game. ANTLR grammar, LSP and VS Code extension.

MusicBox – exports my Ableton Move sessions and plays them in the browser, with bar-quantised clip launching, per-track mixing and DJ-style master effects.

ferret & ExtensibleMigrations – public .NET libraries on NuGet: Postgres-first paging and search; custom EF Core migrations.

Also: GitOps (ArgoCD) k3s home cluster, spec-generated Blazor port of shadcn/ui, event-sourced offline-first apps.

SKILLS

Languages: C#/F.NET, Go, TypeScript, SQL, Python, Kotlin, Java, Ruby

Backend & architecture: ASP.NET Core, EF Core, .NET Aspire, PostgreSQL, RabbitMQ, MassTransit, Redis, distributed systems, event-driven design, system design

Infra & delivery: Terraform, GCP, Kubernetes, Docker, GitHub Actions, CI/CD, Datadog, observability

Data: BigQuery, dbt, Pub/Sub, Dapper

AI-assisted development: Claude Code, agent skills/plugins, multi-agent orchestration (Orca), MCP, spec-driven development, model tiering/cost control

Frontend: Next.js, React, TypeScript

Ways of working: code review, mentoring, technical leadership, developer experience

Head of engineering reference available on request.

EXPERIENCE, CONTINUED

2022 - 2024

Senior Software Engineer – Improbable Worlds

Promoted from Software Engineer. Distributed-systems and metaverse technology.

Platform Operations & Web Platform

- Built and operated an end-to-end **telemetry and analytics platform** over live events (MLB, Yuga Labs), across web, Unreal and backend sources: per-project Terraform, time-partitioning, and a dbt/Airflow → marts → Looker Studio pipeline several teams self-served from.
- Towards the end, led the company's data and intelligence effort without the title: set direction and drove the telemetry build-out and analytics processing.
- Productised a company side project for NFT indexing and tooling, and took on management of company infrastructure through Terraform.
- Line manager for several developers, managing and mentoring them in their careers (including two juniors and a career-changer from an unrelated discipline); led a workstream through one of them.

IT Engineering

- Built internal tooling in Go, Terraform and Kubernetes: joiner-mover-leaver software, Slack-driven Perforce user management, DocuSign and Google Workspace integrations, with CI/CD for each. **Promoted to Senior** on this work.

2020 - 2022

Developer – Wushu Studios

- Unreal Engine C++ on the UI refresh of PS5 *Destruction AllStars*, including live-service work; root-caused team-blocking issues.
- Owned studio IT and infrastructure: a Go Slack bot that became the IT front door, Grafana and Prometheus monitoring managed by Ansible. Hired and onboarded the first two dedicated IT staff.

2016 - 2020

Server Developer – vTime

- Developed and maintained the Java (later Kotlin) backend for a social VR/AR platform: geolocation search, generalised permissions and notifications, realtime messaging and Kinesis analytics ingestion.
- Led the move from SVN to Git/GitLab with better CI and Gradle builds, negotiating concessions for long-tenured artists on legacy tools; it stuck. Cut test runs from 1h20m+ to under 20 minutes.
- Built Vue admin apps and integrations with Mailchimp, AWS SNS and Apple IAP. Mentored a developer from a non-engineering background.

2012 - 2016

Earlier roles

Software Engineer – Faria Education (2016). Full stack on student submission systems in Rails, AngularJS and JRuby on AWS.

Web Developer – Common Space (2016, contract). PHP plugins and Docker-based deployment.

Linux Support Engineer – Bytemark Hosting (2012-2016). Wrote the software around the job: monitoring and triage dashboards the team kept extending, ticketing-system extensions and service tooling, alongside 24-hour on-call and **st**race-level diagnosis. LINX LAIT-1 (2013).

EDUCATION

2008 - 2012

BSc (Hons) Computer Science, First Class – Keele University

Technical lead and treasurer of the student radio station.