Dean Norman

Senior Engineer · AI Engineer

15+ years building software | Cape Town, South Africa | South African citizen | Remote or hybrid

Notice: 0 weeks from the start of a calendar month

dnorman88@gmail.com github.com/DeanNorman

ABOUT

I ship AI into real products, where the model's output has to hold up in front of the people who use it. In October 2026 I built Tenderly, a lean AI tool my team now uses to pick which government tenders are worth bidding on: **the AI reads, code checks every quote against the document, and a person decides.** I choose models by blind test, count in code how much the model actually saw, and switch learning on only once a test proves it helps. Behind that sit 15+ years of product engineering across fintech, agency delivery, delivery leadership and test automation: operations dashboards, backend platforms and internal tools that have to be right.

I joined Zamatrack to build one app, and over the following year grew into the engineer its platform's delivery runs through: one screen, one release and one decision at a time.

SKILLS

CORE	React · TypeScript · JavaScript · React Native · Node.js · PHP · PostgreSQL · MySQL · Next.js · Playwright · Jest
AI	LLM integration · Gemini · Claude Haiku · Structured output · Constrained decoding · AI evaluation by blind test · Validation with Zod and Pydantic · Human decision in the loop · Agent-assisted delivery with Claude Code · Checkout for AI agents (UCP)
ALSO	Python/FastAPI · Jotai · TanStack Query · Leaflet · Prisma · SQLite · Redis/BullMQ · Docker · Cloudflare Workers and D1 · Vitest · pytest · Storybook

WORK EXPERIENCE

Senior Engineer · AI Engineer · Zamatrack

OCT 2025 TO PRESENT · CONTRACT · REMOTE

Zamatrack builds Worxit, the software behind physical security and facilities operations: guarding companies, residential estates and the control rooms that run them. I joined to build the resident app, then grew into the dashboard's architecture, testing and releases, then the PHP backend and the delivery route. **I am the primary author of Worxit's operations dashboard, resident app and tender tool, and since August 2026 the main engineer on its PHP backend.** I run the release process and review and ship my colleagues' features.

- **Asset management: the largest piece of work I have delivered here, now a product area in its own right.** Equipment (radios, cameras, vehicles, alarm hardware) spread across client sites lived in spreadsheets nobody could answer questions from. The system tracks what the company owns, where each item is, who is responsible for it and what condition it is in. Reported faults become the same kind of job operations teams already handle. Site audits run on the dashboard and lock once a manager signs. Finance sees current value computed from purchase price and a depreciation profile, with disposals fixing gain or loss at the moment of sale and every valuation report carrying the method behind its own figures. Scheduled servicing generates ordinary work orders, so none of it needed a new mobile release to reach the field. Shipped as five sequenced phases across the dashboard and backend, adding three new product areas.
- **Operations dashboard: what a control room uses to run a shift.** Every site on a live map, with the jobs needing a response in a panel beside it that cannot be closed away, so work waiting on someone stays visible instead of scrolling out of a feed. Alarms needing action, routine site movement, and devices that have simply stopped reporting are three separate things; operators had been reading them as one noisy stream. I led the product decisions, the interface and the language: the team works from a shared glossary so support, sales and engineering describe the same thing the same way.
- **Modernised it without stopping it.** Replaced Recoil with Jotai, and moved server data onto one TanStack Query layer that pauses polling while the tab is hidden. Gave the dashboard one shared request client: a lint rule now rejects any new one-off network call, and the roughly 250 older ones move over whenever their file is touched. Analytics show only numbers computed from real data, with invented fallbacks and always-green status pills removed. Added a Storybook design library kept apart from the live app, one command palette for both operator screens, and an incident replay that rebuilds what happened on the same map and steps aside the moment a live incident needs the operator.
- **Resident access control mobile app.** What residents of a gated estate use: verifying their device, inviting guests so security knows who is coming, booking shared facilities, receiving notices from management. Built to app-store submission standards and for bad connections: no request can hang, and bookings never queue offline, because the slot may be gone when the signal returns. Not yet in the app stores. I led its resident access feature across the backend, dashboard and app: built and proven on production behind a per-company switch, not yet switched on for any company.
- **Cloudflare and platform work, one checked step at a time.** I audited the Cloudflare account read-only before changing anything, then worked through the findings in ticketed, verified steps: moved the dashboard onto Cloudflare with builds straight from Git (a merge goes live within minutes; the marketing site now deploys automatically on merge too), moved a related company's website onto Cloudflare, retiring a manual FTP deploy and moving its contact form into the edge code after the old host blocked outbound mail, wrote the firewall rule so web traffic reaches the new servers only through Cloudflare, put mail backups on locked storage with a tested restore, and built the self-hosted mail server and migrated staff mailboxes to it. I kept to free-tier options and took anything that carried a cost to the owner as a written decision.
- **A checkout AI assistants can use.** On the company's online store, an assistant finds the products, builds an order priced with VAT and delivery, and hands the buyer a link to review and place it; payment and the final decision stay with the person. I built it in phases, each with a written spec first, and tested it end to end with an AI agent given only the task.

- **Made a risky backend safe to release.** A live multi-tenant PHP API with a separate database per client, no test copy to rehearse on, and deploys done by pasting code into a web terminal. I first built a release lane that shipped one change at a time, each with a written plan, a dry run against the live server, automatic checks and a backup to roll back to. Then I led the move to one route for the whole team: a merge to main is now the only way a change reaches production. I also built a one-command local backend, and every change I ship is built and tested on it first.
- **Identity and compliance checks.** Added identity and compliance checks to the operations platform: ID verification, sanctions and PEP screening, and bank account and company checks, with bundled KYC checks gated behind a lawful-basis step.
- **Tender screening.** Government tenders arrive as a pile of PDFs, spreadsheets and scans, and someone has to decide which are worth bidding on. Worxit Tender pulls new tenders from the government feed, reads the documents whatever format they are in, and produces a short brief with the deadline, the requirements and anything disqualifying, with every fact linked back to the page it came from. The decision I would defend is keeping four questions apart: does this fall in our field, can we do the work, is it worth bidding, and what did the person decide. In October I built Tenderly, a lean MVP on a sharper domain model: AI reads, code checks every quote, a person decides. Our team now works from it.
- **Reviews, quality and AI-assisted delivery.** I review and ship my colleagues' features: structured reviews, blocking issues fixed before release, and every release shipped with a written test record. Automated tests back the work, including tests that drive the real interface the way a person would and integration tests that run against a real database rather than stand-ins. AI assistance works inside the same ticketed, human-approved workflow as every other change, and I built the AI skills the team uses to produce branded training notes, technical records and decks, with automatic checks on layout and screenshot freshness.

SKILLS `React` `TypeScript` `React Native` `Node.js` `PHP` `PostgreSQL` `Next.js` `Playwright` `Jest` `Prisma` `Gemini` `Cloudflare`

Test Automation Consultant · Kepler + Co

MAR 2025 TO JUL 2025 · CONTRACT · REMOTE

Short-term contract focused on release confidence. Designed and built end-to-end Playwright suites, automated the key manual regression packs in Tuskr, and cut manual testing overhead so releases shipped faster and more reliably. Worked with the technical team to embed a practical testing culture and improve QA workflows.

SKILLS `Playwright` `JavaScript` `React` `Docker` `Git` `Jira` `Tuskr`

Software Product Manager · Whitehart

JAN 2024 TO AUG 2024 · FULL-TIME · REMOTE

- Led delivery across multiple web projects with a team of developers: scope, planning and timelines.
- Translated messy business requirements into clear, buildable work, and shaped UX and site structure.
- Ran the delivery process and stakeholder communication; kept quality high through review, testing and documentation, delivering mostly WordPress projects from brief to launch and improving turnaround by tightening scope and review.

SKILLS `Scrum` `Agile` `Jira` `WordPress` `Product roadmaps` `Product strategy` `UX design` `Stakeholder management`

Software Manager · Silicon Overdrive

JAN 2021 TO NOV 2023 · PART-TIME · REMOTE

Led software delivery across multiple client projects. Planned scope, timelines and resourcing, and acted as Scrum Master for day-to-day delivery. Ran client communication and expectations, mentored junior developers, and kept documentation and reporting standards, delivering projects of varying complexity from inception to completion under deadline pressure.

SKILLS `PHP` `Laravel` `CodeIgniter` `WordPress` `React` `AWS` `Figma` `Scrum` `Notion`

Scrum Master · Enabled World

SEP 2021 TO OCT 2021 · CONTRACT · REMOTE

Short engagement on an instant-messaging platform. Joined as Scrum Master and quickly moved into hands-on React contribution alongside delivery coordination: real-time messaging product work, international messaging behaviour on legacy devices, and testing and performance measurement on a live communications platform.

SKILLS `React` `Agile` `Jira` `Trello` `Notion` `Confluence`

UX / Frontend Developer · 22seven Digital (Old Mutual)

SEP 2014 TO MAY 2021 · FULL-TIME · HYBRID

Almost seven years on Old Mutual's personal-finance app, working where product engineering, UX and user-behaviour data met.

- **Product engineering.** Built and maintained features on the core money-management app (JavaScript MEAN stack, Sass, in-house responsive frameworks), built React components for the web app, and worked on the React Native implementation. Looked after the public site (custom PHP), blog (WordPress) and support-centre theme from local dev through staging and production, including cross-browser testing.
- **UX and product discovery.** Ran Google Design Sprints and rapid prototyping (Figma, Sketch, InVision, proto.io) and facilitated user testing on real financial-behaviour problems, iterating designs with the UX team from behavioural and analytics findings.
- **Data and growth.** Queried Amazon Redshift (SQL), Google Analytics and Excel to turn raw extracts into insights that directed features, UX, communications targeting and KPI tracking; technical SEO, analytics audits and agency-partnered content and email campaigns.

SKILLS `JavaScript` `Angular` `Node.js` `MongoDB` `React` `React Native` `PHP` `WordPress` `SQL` `PostgreSQL`

Partner · The Blog Boat

JAN 2013 TO FEB 2016 · PART-TIME · REMOTE

A niche WordPress and Blogger theme shop created by my partner; I joined as a partner soon after launch. Sold solely on Etsy and run in our spare time: theme design and build, product listings, customer support.

SKILLS `WordPress` `Photoshop` `Illustrator` `Kanban`

Web Developer · Urbian

NOV 2012 TO AUG 2014 · FULL-TIME · OFFICE

Digital agency work for brands including Captain Morgan, Windhoek, Lindt and Drive Dry. Started in interface design and campaign updates, then moved into frontend development: scoping and technical specifications, HTML5/Sass/JavaScript, MVC frameworks (Backbone, Laravel, CodeIgniter), responsive builds on Foundation and Bootstrap, cross-browser delivery, and interface design for native Android.

SKILLS: HTML5 · Sass · JavaScript · PHP · Laravel · CodeIgniter · Backbone.js

Web Designer · Freelance Web & Graphic Designer

JAN 2009 TO JAN 2012 · PART-TIME · REMOTE

Ran a small freelance operation with my partner through college, building websites for small businesses, mostly WordPress and PHP. Early practice in client communication, translating technical concepts for non-technical people, and meeting deadlines.

SKILLS: WordPress · PHP · HTML · CSS · jQuery · Photoshop · Illustrator

PROJECTS

Tenderly · picking which government tenders are worth reading · Zamatrack

OCT 2026

PROBLEM Thousands of government tenders are open at any time, in PDFs, scans, Word and Excel files, and the team has to pick the few worth reading and bidding on. After building Worxit Tender, a company-sized platform that is still in service, I saw a sharper use case and a better domain model for that daily job.

IMPACT Built overnight as a lean MVP, alongside Worxit Tender rather than replacing it, on a new codebase and a fraction of the server code. AI reads, code checks every quote, a person decides. Our team now works from it.

HOW I USED AI An AI gate scores the open listings and sets most aside; an AI reader reads the best twenty, scans included. Code checks every quote against the document, and each analysis shows how much was read and a set of critical checks. A person decides Bid, Maybe or No with a one-tap reason. A blind test against my own decisions chose Claude Haiku for the gate, and a decision memory is built but switches on only once a replay proves it.

TECH: TypeScript · React · Node.js · Express · SQLite · Zod · Gemini · Claude Haiku · Vitest

Worxit Operations Dashboard · production physical-security operations · Zamatrack

2025 TO PRESENT

PROBLEM Operators needed control-room confidence without urgent response work, routine estate movement and device reliability blurring into one noisy interface.

IMPACT A dashboard and live map split, a permanent response lane for owed work, separate states for attention, routine movement and reliability, polling-aware state management, and analytics where a number renders only if it is computed from real data. The same model later carried the asset work: lifecycle and faults, signed site audits, custody and documents, then computed and versioned valuation. One decision kept it from sprawling: faults and recurring maintenance both feed the ordinary work orders the teams already run.

HOW I USED AI Agent-assisted delivery with Claude Code inside the same ticketed, reviewed workflow as every change, working from a shared domain glossary and written decision records.

TECH: React · TypeScript · Leaflet · Jotai · TanStack Query · Rsbuild · Storybook · Jest · Playwright · PHP API

AI Content Workflow Engine · independent AI-assisted creator workflow

2026

PROBLEM Creators needed a repeatable way to move from competitor signals to reviewable campaign assets without blank-page friction or brittle one-off prompting.

IMPACT Turns competitor Instagram signals into a reviewable content package (insights, slide copy, generated assets, caption, hashtags and ZIP/TXT handoff files), leaving publishing with the creator. The product is the handoff, not the generation.

HOW I USED AI It began with several LLM providers and a fallback chain; since July 2026 it runs on Gemini alone. JSON parsing safeguards, Pydantic validation, near-duplicate filtering, brand-voice context and deterministic SVG/template generation keep it useful when generation is imperfect.

TECH: React/Vite · Python/FastAPI · PostgreSQL/SQLAlchemy · Pydantic · Docker · Playwright · pytest · Gemini

AI FLUENCY

I use AI in my workflows

AI is part of how I deliver, not a side experiment. I use Claude Code for implementation, review and documentation, with work broken into ticketed steps, a stated check for each change, and a written decision record for anything a future maintainer would otherwise have to guess. Repository evidence and human verification decide what is accepted.

I ship AI features

On Worxit Tender I built the AI layer end to end: Gemini extraction behind a validated-fact firewall, so generated prose may only state facts already recorded from the source, never a recommendation or a score. Constrained decoding against a slimmed response schema keeps output parseable, and priority bands put on-demand requests ahead of background work inside a bounded daily budget. On Tenderly, AI reads, code checks every quote and a person decides.

I measure before I trust

Tenderly's models were chosen by a blind test against decisions I had made myself, with the pass mark agreed before the run so I could not tune to the answer. Every disagreement traced back to the hand-written profile of what we sell, not to the model: the profile matters more than the model. Coverage is counted by code, which exposed a length limit silently cutting long tenders short.

Where I apply extra caution

Output that passes a schema can still be wrong, so it is checked before anything downstream relies on it: strict validation, quotes checked against their source, and a person making every decision. Anything that would learn from past decisions stays switched off until a replay proves it beats the hand-written profile.

INDEPENDENT TOOLS

CopyPasta, an open-source (MIT) browser extension, live on the Chrome Web Store, restoring native copy, cut and paste on sites that suppress clipboard events: inert until enabled per site, no clipboard reads, no network calls, no telemetry, zero runtime dependencies. `JavaScript · Manifest V3 · node:test · github.com/DeanNorman/CopyPasta`

Local Asset Converter, a native SwiftUI macOS workspace for local video, audio, image, SVG and frame-sequence conversion, with inspectable generated commands and non-destructive output handling. `SwiftUI · FFmpeg · XCTest`

Spotify Autocapture, a Go/Bubble Tea TUI that infers when playback leaves the album or playlist you chose and routes discoveries to a reviewable deck, keeping history and tokens local. `Go · SQLite · Spotify API`

EDUCATION

Bachelor of Arts, Creative Brand Communications (Multimedia Design) · Vega College, Cape Town `2010 TO 2012`

National Senior Certificate · Elkanah House `2002 TO 2006`

Some of this work is private or commercially sensitive. No client data, tenant identifiers, operational locations, device names or screenshots are included here; architecture, decisions and sanitised implementation evidence can be walked through in conversation.