SOFTWARE ENGINEER

Peter Wilkins

Functional backend engineer for complex regulated domains.

I turn complex domains into simple, reliable software: payment systems, billing workflows, immutable data models, executable rules, local-first tools and AI-augmented engineering systems.

My strongest work is usually at the boundary between domain understanding and architecture: finding the durable facts, separating concerns, making trade-offs explicit, and keeping systems understandable as they grow.

LOCATION
Wyke Regis, Dorset, UK

EMAIL
peter.wilkins2@protonmail.com

GITHUB
github.com/peter-wilkins

WHAT I BRING

Complexity into working systems

Domain modelling	**Functional design**	**Regulated systems**	**AI leverage**

PROFESSIONAL EXPERIENCE

Backend systems in finance, energy, healthcare and SaaS

JUXT

Software Engineer / Consultant

February 2022 - July 2025

Consultant building cloud-native backend systems for large financial organisations using Clojure and functional programming.

- Worked on Funding Circle payment workflows involving Direct Debits, Faster Payments and customer communications.
- Built services for regulated financial environments where correctness, resilience and operational safety mattered.
- Delivered backend services across Clojure, Python and Kotlin with Kubernetes, Terraform, AWS and Docker.

OVO Energy

Software Engineer

April 2020 - February 2022

Worked across payments and billing modernisation, including event-driven architecture and schema evolution.

- Developed payment services supporting Direct Debits, Faster Payments and customer communications.
- Helped redesign billing around asynchronous event streams rather than tightly coupled push workflows.
- Contributed to schema evolution strategies that reduced deployment risk and simplified long-term maintenance.

Polecat Intelligence

Software Engineer

January 2018 - April 2020

Built core components of an intelligence platform using Clojure, Datomic, GraphQL, Kafka and Elasticsearch.

- Designed and built a taxonomy management system using Clojure and Datomic.
- Developed a GraphQL API that mapped naturally onto Datomic's immutable data model.
- Used Datomic's temporal query capabilities to identify changes efficiently for downstream processing.

Dickies

Software Developer

July 2017 - January 2018

Backend development for a large e-commerce platform.

Mayden

Software Developer

September 2015 - July 2017

Developed patient-management software for NHS mental health services.

ARCHITECTURE THEMES

Recurring design decisions

Local-first before cloud sync where user trust, offline work or data loss matters.	Append-only events and immutable records where recovery, auditability or derived status matters.
Small deterministic cores with flexible outer workflows.	Functional boundaries and explicit data shapes for complicated domains.
AI as reviewer, planner, operator and research partner, not just as a code generator.	Human-in-the-loop decisions where automation would be risky or premature.

Recent projects as architecture evidence

INSPECTABLE AUTOMATION

Commandbook

DESIGN DECISION

Model commands as shared, inspectable recipes with named inputs, outputs, side effects, dry-run behaviour and trust expectations. Resumable coffee-grinder runs can pause for missing facts, human choices, permissions or platform failures.

PHYSICAL PRODUCT WORKFLOW

Shiny Art Shop

DESIGN DECISION

Model each order as a Project whose status is derived from facts and events. Separate customer, workshop, admin, payment and system commands. Use provider-swappable preview generation, with deterministic ImageMagick rendering as a stable comparison path for AI image experiments.

EXECUTABLE RULES

RegenOS

DESIGN DECISION

Model grants as structured tasks, eligibility rules, evidence requirements, claims and lifecycle gates. Keep country-specific schemes as data rather than hard-coding one government process.

AI SPEECH PIPELINE

WhisperWayland

DESIGN DECISION

Treat transcription as a pipeline: capture, VAD, audio conditioning, streaming chunks, provider racing and text cleanup are separable stages with measurable trade-offs.

JobDone

Local-first operational memory for work: captures stay local until confirmation, confirmed entries are immutable, and sync is treated as a replica rather than the user's source of truth.

Continuum Core

A source-log and resume-brief layer for continuity of thought: the user should not have to maintain the filing system before the system can help them re-enter context.

Foil Board Toolkit

A parametric generator rather than a clone library: study public design trends, but produce original inspectable board definitions, geometry reports and generated review artifacts.

CNC Workshop Tools

Conservative digital-fabrication tooling: prove geometry and toolpaths in simulation, keep manufacturing assumptions explicit, and separate visual generation from machine-ready validation.

AI ENGINEERING

Human-agent engineering

Most of my recent work was built in collaboration with AI agents. I do not treat agents as magic code generators or autonomous replacements for engineering judgement. The useful pattern is symbiosis: agents accelerate research, implementation, test generation and review; I provide product direction, architectural taste, domain judgement, acceptance criteria and the decision about what is actually worth building.

I could not move this quickly without AI. The agents would not produce coherent products without human direction. The value is in the working relationship.

ENGINEERING PHILOSOPHY

Simple enough to reason about

My engineering style is strongly influenced by Rich Hickey's emphasis on simplicity, explicit design and separating concerns. I enjoy software that makes difficult domains understandable rather than hiding them behind layers of accidental complexity.

Good AI tooling makes this more important, not less. When implementation gets faster, the limiting factor becomes judgement: choosing the right surface area, knowing what not to build, and keeping the system coherent.

TECHNICAL SKILLS

Tools I use to build these systems

MAIN LANGUAGES

Clojure, Scala, Kotlin, Python, JavaScript, SQL

FUNCTIONAL AND DATA SYSTEMS

Datomic, Kafka, event sourcing, immutable records, schema evolution

CLOUD AND PLATFORM

AWS, Kubernetes, Terraform, Docker, Linux

DATABASES AND APIS

PostgreSQL, GraphQL, Elasticsearch, REST

AI ENGINEERING

Agent workflows, LLM review loops, transcript pipelines, AI-assisted prototyping

CONTACT

Get in touch

Email peter.wilkins2@protonmail.com or see recent public work at github.com/peter-wilkins.