Patryk Gniewek

Katowice, Poland · open to remote · patrykgniewek9@gmail.com · +48 691 108 952 · github.com/ptsik

SUMMARY

Fullstack engineer with 2+ years building production backends in TypeScript and Node.js, with shipped production work in Python and Rust. I own features end to end - data model, event pipelines, APIs and UI - and specialise in correctness under unreliable conditions: out-of-order events, idempotent writes, retry classification and self-healing reconciliation. Two years working remotely with distributed teams in the US and Japan. Currently running my own practice alongside a launched SaaS product with paying customers.

EXPERIENCE

Fullstack Engineer (freelance) - Kaijiro Labs

Mar 2026 - present

Katowice, Poland · remote · own product plus client engineering

- **Marisol** - bilingual (EN/ES) AI voice receptionist, launched solo: real-time voice agent, Google Calendar booking with double-booking prevention, SMS confirmations and Stripe subscriptions. **6 paying customers.**
- Four client projects delivered: a book sales site with Autopay payment integration (Next.js, Three.js, Cloudflare), two promotional sites, and a tutoring platform currently being extended into a full e-learning product.

Backend Engineer - Candylabs

Oct 2025 - Feb 2026

Tokyo, Japan · remote · 4-person engineering team · team wound down

NFT marketplace on Solana - 8,700+ users, ~50,000 transactions, 42% returning. I owned the indexing and data layer

- **Built the platform's event ingestion pipeline from scratch** - streaming ingestion from an external source, schema-driven decoding, reconciliation jobs, and Prometheus instrumentation with a 5s write-delay SLO and p95/p99 histograms.
- **Designed and built the shared database package from scratch**: schema, indexes, enums and validation for every service touching marketplace data.
- **Built the backfill system that closes the pipeline's delivery gaps** - a scheduled sweep trailing the live stream to avoid races, primary provider with fallback and backoff, signature-level deduplication, and Redis checkpoints so a restart never loses progress.
- Designed ordering guarantees for a source that delivers events **out of order and duplicated** - conditional, version-checked writes stopped late events from reverting newer state.
- Built the precomputed statistics layer after the product's largest launch (**66% of lifetime volume**) exhausted the Postgres connection pool: background refresh every 2-5 min with jitter plus a batching write queue (flush 1s / 50 ops). A multi COUNT(DISTINCT) query across two tables became a single primary-key JOIN.
- Added signature-based (ed25519) upload authentication and the first unit tests the main API ever had - **29 tests across 8 services**, gating every merge in CI.

Fullstack Developer - early.cash / Waterfall

Dec 2024 - Oct 2025

New York, USA · remote · 4-person engineering team · team wound down

Token fair-launch platform - 269 commits across 3 generations of the codebase, owning refunds, indexing and on-chain security · Rust, Python, TypeScript

- **Built the refund pipeline**: maturity delays measured in chain slots rather than wall-clock seconds, so amounts were computed on stabilised state; a Redis sorted set as the queue; per-wallet aggregation of contributions for failed launches.
- **Designed the identity scheme** that replaced a reusable public name as the record key, ending **silent overwrites of earlier records' financial history**. Migrated the API, five Lua scripts and two workers without changing public URLs; the team later extended the scheme on-chain.
- **Rebuilt the external RPC layer** - batched fetches cutting N HTTP round-trips to 1 with recursive retry of only failed entries, typed deserialisation replacing hand-indexed JSON, and retryable-vs-fatal error classification. This surfaced a defect recording failed transactions as settled.
- Secured four fund-moving on-chain instructions (Rust) against invocation from external programs, synchronising the program, its interface definition, the Python client and the frontend.
- Rewrote the swap feature onto a typed client generated from the interface definition, removing two heavy legacy SDK dependencies (+2,783 / -500 lines).
- Built observability from scratch (Loki, Grafana, Sentry, dedicated alerting service) and added Brotli compression with content negotiation to long-lived Server-Sent Event streams.

Frontend Developer (contract) - Arkitekt DEX May 2024 - Sep 2024

Remote · product discontinued

- Implemented the trading UI from design specs - swap, liquidity pool and staking views - plus wallet connection flow and navigation.

SELECTED PROJECTS

automation-toolkit - Claude Code plugin

- 10 AI agents, 17 skills and 10 MCP integrations with a confidence-gated review step before deploy. Daily driver for client work.

SKILLS

Core languages: TypeScript, JavaScript, SQL

Backend: Node.js, Express, Hono, NestJS, REST, WebSocket, Server-Sent Events

Data: PostgreSQL, Redis (Streams, Lua), MySQL, MongoDB, Drizzle ORM

Queues & events: BullMQ, Redis Streams, Kafka, idempotency and retry patterns, dead letter queues

Frontend: React, Next.js, Remix, Tailwind, Zustand, TanStack Query, Three.js

Infrastructure & observability: Docker, AWS, Cloudflare Workers, GitHub Actions, Prometheus, Grafana, Loki, Sentry

Testing: Vitest, Jest, integration testing, Swagger/OpenAPI

AI tooling: Claude Code, Cursor, MCP, multi-agent workflows

Working knowledge: Python (FastAPI, async workers), Rust (Solana/Anchor programs), EVM/viem - shipped to production, not my primary languages

EDUCATION

BEng Computer Science - Internet & Mobile Technologies 2020 - 2024

University of Information Technology and Management (WSIiZ), Rzeszow, Poland

LANGUAGES

Polish - native · English - C1, daily working language for 2 years across US and Japan based teams