JOSHUA OKECHUKWU

Senior Backend Engineer: Go, Payments and Settlement

joshuaokechukwu001@gmail.com | linkedin.com/in/joshuaokechukwu001 | github.com/Parallax18

PROFESSIONAL SUMMARY

Senior backend engineer with 6+ years building and running systems that move money. Go is my primary language, and I have owned payment and settlement infrastructure end to end: provider integrations, payouts, ledger accounting and reconciliation at a digital asset exchange and a real-time payments platform. I build for correctness under failure. Idempotency, row locking, retries and reconciliation are how I stop duplicate webhooks, chain reorganisations and provider timeouts from turning into lost funds. AI-native: I run coding agents daily as a working tool, and I treat their output as a hypothesis to verify against code, provider documentation or a live call.

TECHNICAL SKILLS

- **Languages:** Go (primary), TypeScript, Node.js, Rust, SQL
- **Payments and settlement:** payment provider and banking API integrations, payouts, settlement, reconciliation, webhooks and callbacks, double-entry ledger accounting, on and off-ramp integrations, KYC provider integration (BVN and NIN identity checks)
- **Correctness under failure:** idempotency keys, balance holds, SELECT ... FOR UPDATE row locking, fixed lock ordering, partial unique indexes, retries with backoff, outbox pattern, at-least-once delivery, duplicate suppression
- **Data:** PostgreSQL (schema design, indexing, query tuning, transactions), Redis, GORM, pgx, Prisma, MongoDB, SQLite; migrations and backfills
- **Redis in production:** caching, atomic Lua scripts, token-bucket rate limiting, single-use signed intents, queue-backed jobs (BullMQ)
- **Cloud and infrastructure:** AWS (S3, IAM; services deployed and run on AWS), Docker, Compose, private networking, self-hosted key management service, Vector log shipping, Dokploy CI/CD deploys, Linux VPS operations
- **Observability:** Sentry with PII scrubbing, PostHog, structured JSON logs, request IDs, liveness and readiness probes, log pipelines, production debugging and incident response
- **AI-native workflow:** Claude Code, Codex and OpenCode (in Zed) for implementation, test generation, code review, documentation and debugging; repo-level agent context files and written decision logs

PROFESSIONAL EXPERIENCE

Founding Backend Engineer, Riddo (digital asset exchange) · *May 2025 - Present*

- Owned the Go backend end to end: domestic bank transfers, crypto deposits and withdrawals, a double-entry asset ledger, payouts and reconciliation. 29 modules, around 42k lines, 39 test files including money-path integration tests.
- Worked with an independent cybersecurity firm to audit the platform for technical loopholes and policy violations, then drove the remediation through to production. I ran a module-by-module sweep of money integrity and failure modes alongside it, fixing replayable withdrawal intents, double-settle races and silently swallowed provider errors, each with an integration test that fails without the fix.
- Integrated a banking-as-a-service provider (accounts, transfers, callbacks), an identity and KYC provider, a crypto custody provider and blockchain node providers, building for their failure modes rather than assuming success responses.
- Designed an append-only, double-entry ledger with reserve and asset accounts and debit, credit, reserve and reversal entries, so every balance is derived from entries instead of being mutated in place.
- Made money paths safe under concurrency: SELECT ... FOR UPDATE row locks, a fixed Reserve-before-Asset lock order that removes deadlock, partial unique indexes for idempotency, and a balance hold taken before any transfer fires and released if it fails.
- Cut asset-to-cash settlement from hours to under two minutes by replacing synchronous provider calls with a queue-backed pipeline over the fiat rails.
- Built reconciliation that compares provider reserve balances against ledger bookkeeping and sweeps payouts stuck on missing webhooks, with no funds lost or double-paid across concurrent settlement.
- Built a separate blockchain gateway service (TypeScript, Hono, SQLite): raw webhook bodies stored before parsing, HMAC-verified ingress, an outbox with signed at-least-once delivery, four-eye approval before any key signing, and a boot-time guard that refuses to start when the process network disagrees with the database.
- Built device-bound authentication on Secure Enclave keys, TOTP 2FA with argon2id hashing, and fine-grained RBAC with audit logs for admin operations.
- Set the repo's engineering standard: verification commands before every commit, one logical change per commit, provider documentation over our own assumptions. Wrote the architecture map and decision log the project runs on.

Founding Backend Engineer, Fanpool (real-time payments platform) · *Dec 2023 - Jun 2025*

- Solo-built and ran a real-time platform from zero to 2,000+ users and thousands of dollars in transactions, owning design, delivery and production support.
- Integrated Circle developer-controlled wallets on Polygon and a Scalex NGN/USDC ramp: wallet creation, HMAC-SHA256 signed provider requests, transfers, and webhook state machines covering onramp, offramp, completion and failure.
- Built money movement across several rails: USDC on Polygon, NGN bank transfers through payment links, an internal token bank with its own reserve and fee wallets, and a bill-payment marketplace for airtime, data, electricity and vouchers.
- Built queue-backed payouts, scoring and notifications on BullMQ and Redis with retries, idempotency keys and duplicate suppression, so a retried payout cannot pay a user twice during peak game weeks. Competition kickoff, settlement and quiz sessions ran as delayed jobs keyed by job id, so a duplicated schedule could not fire twice. Exposed queue health through Bull Board.
- Used Redis for caching and job state, MongoDB with Prisma for transactional data (interactive transactions for multi-step writes), and WebSockets with JWT-authenticated connections for live updates.
- Ran production on AWS (S3, IAM) and a Linux VPS with structured logging, monitoring and crash recovery (BetterStack): over 99.9% uptime during game weeks and under two minutes to detect an incident.
- Integrated the official FPL data API with credential-based account linking and a daily sync job, a differentiator that cut churn by roughly 30%.

Frontend Engineer, Clearerpay (payments) · *Mar 2024 - Nov 2024*

- Delivered core frontend functionality for a payments product, keeping the flow stable through major team turnover.
- Integrated the frontend against internal services and external payment partners, covering error states, retries and recovery from failed requests.

Frontend Engineer, Mainstack · *May 2023 - Dec 2023*

- Owned the frontend for the Invoicing, Mediakit and Automation modules, integrating the backend services and REST APIs behind them.
- Built the Automation module around an event-driven model, where user-defined triggers dispatch asynchronous jobs with retry and duplicate suppression.
- Built reusable form handling and validation logic that removed a whole class of repeated defects across modules.

Full-Stack Engineer, Independent client projects · *Feb 2020 - Present*

- Client projects in Nigeria and the UK: CMS-backed marketing platforms, Next.js SEO tooling, and API and frontend performance fixes.

HOW I WORK WITH AI

- Run an agent-first workflow with Claude Code, Codex and OpenCode inside Zed for implementation, test generation, review, documentation and debugging, backed by repo-level context files that encode build, test and safety rules so agents work inside the real constraints of a money system.
- Design before code: plain-English specs with options and trade-offs, plus a decision log that records why each choice was made, so humans and agents share one map of the system.
- Commissioned an AI-run production-readiness audit across the whole backend, then verified every finding line by line against the code. That included retracting a plausible root cause the evidence disproved.
- Keep the tools honest: no money-path change ships without a test that fails without it, a check against the provider's own documentation, or a live call.

ADDITIONAL

- Based in Nigeria. Fully remote, and happy to overlap with UK and EU hours.
- Available at short notice, and open to contract with conversion to full-time.