RESUME · BACKEND

# John Kingola @Astralchemist

Backend engineer · Python, FastAPI, PostgreSQL and TimescaleDB · Multi-tenant SaaS and rate-limited APIs

> I check that something is true before anyone relies on it: trading strategies, AI agents, and the systems around them.

Nairobi, Kenya (UTC+3) | remote as a contractor or through an EOR | no visa sponsorship needed

I design and ship backend systems end to end. I built the multi-tenant Change Intelligence Platform solo in six months (Next.js, FastAPI and Supabase, four role-based tiers, a self-hosted LLM) and still maintain it; designed a backtest-as-a-service API on TimescaleDB with a Redis cache keyed on the strategy hash and per-key rate limits; and at ConsultHasha built the REST layer and production endpoints the product and its client integrations ran on, owning architecture, deployment, and reliability. I have shipped production software for six years, since 2020, and I treat performance as something to measure: the platform's latency pass followed an audit that found I/O-wait bottlenecks and N+1 queries.

github.com/Astralchemist LinkedIn kingolajohn55@gmail.com kingola.dev

| Solo | 4 | 60/min | 6 yrs |
|---|---|---|---|
| built a multi-tenant SaaS, architecture to production | role-based access tiers | per-key API rate limits | shipping production software |

NOW — On retainer for the Change Intelligence Platform: maintenance and feature work on its Next.js, FastAPI, and Supabase stack.

## WHY HIRE ME: DECISIONS, AND HOW I CHECKED THEM

### Why does the same backtest get computed twice?

- THOUGHT — The same strategy is requested a hundred ways, so a cache keyed on the request keeps missing.
- TESTED — Keyed the Redis cache on the strategy hash, with per-key rate limits of 60 a minute in front of TimescaleDB.
- RESULT — A repeat request for an unchanged strategy is served from the cache, however it is phrased.
- PROOF — Private repo, described

### Where is the platform's latency actually coming from?

- THOUGHT — Guessing optimises the wrong layer.
- TESTED — Started the latency and build-performance pass from an audit, which surfaced I/O-wait bottlenecks and N+1 query patterns.
- RESULT — The fixes went after what the audit measured, on a multi-tenant SaaS I built solo and still maintain.
- PROOF — change-tca.com

## AI TOOLS I BUILD WITH

Claude Code · Anthropic | Cursor · Anysphere | Codex · OpenAI

## SKILLS

Data and backend

Python | FastAPI | PostgreSQL and pgvector | TimescaleDB | Redis | RabbitMQ

Delivery

Docker Compose | GitHub Actions CI | Cross-platform credential stores | Release engineering | Latency and build-performance passes | Upstream PRs to quant libraries

Web and product

TypeScript · React · Next.js · Tailwind CSS · Node.js · Supabase

AI and GPU inference

vLLM on CUDA GPUs (Modal L4) · PyTorch, Hugging Face Transformers, Accelerate · KV-cache and GPU memory tuning · Llama 3.1 8B fine-tuning and serving · Claude, OpenRouter, Ollama fallback · Tool-calling loops, embeddings, and RAG

Tooling and desktop

Rust (Tauri, research engines) · Kotlin (Compose Desktop) · MCP servers · npm publishing · SQLite · Linux and VPS ops

Trading and research

Backtesting engines · Signal extraction · L2 order books, markouts, and adverse selection · Power laws and EVT · MQL5, MT5, and Pine Script · QuantConnect Lean

## EXPERIENCE

Dec 2025 to Present
Dubai, UAE · remote

### The Change Intelligence Platform · Lead engineer (contract), then maintainer

- Built and shipped the platform solo in six months, Dec 2025 to Jun 2026, from architecture to production, under the direction of change-management practitioners. On retainer since for maintenance and feature work.
- Multi-tenant change-management SaaS: Next.js and FastAPI on a Docker Compose stack with Supabase. Fine-tuned Llama 3.1 8B and served it with vLLM on a Modal L4 GPU, behind an OpenAI-compatible API with an OpenRouter fallback.
- Led a latency and build-performance pass after an audit surfaced I/O-wait bottlenecks and N+1 query patterns.
- Four role-based tiers, including a delegated Champion portal with magic-link auth, PDF export via Puppeteer, a seven-tag sentiment model, and an automated LinkedIn outreach pipeline.

Next.js · FastAPI · Python · Supabase · PostgreSQL · Docker Compose · Llama 3.1 8B · vLLM · Modal · OpenRouter · Puppeteer

Oct 2025 to Present
Dubai, UAE · remote

### Early-stage stealth startup · Technical advisor, algorithmic systems

- Joined as lead algorithmic engineer at the founding stage, then moved to an advisory role while funding closes.
- Designed the backtest-as-a-service API: FastAPI, TimescaleDB, a Redis cache keyed on the strategy hash, and per-key rate limits.

Python · FastAPI · TimescaleDB · Redis

2024 to 2025
Nairobi · hybrid

### ConsultHasha · Principal systems engineer

- Early engineer at a startup. Designed and built the REST API layer and the production service endpoints the product and its client integrations ran on.
- Owned architecture, deployment, and reliability.

2020 to 2025
Remote

### Upwork · Freelance quant and full-stack developer

- MQL5 and MT5 Expert Advisors built to client risk specifications, then Python backtesting tools, Telegram signal bots, and execution dashboards.
- Also Docker-first Next.js and Postgres client platforms.

MQL5 · MetaTrader 5 · Python · Telegram bots · Next.js · PostgreSQL · Docker

## EDUCATION

Sep 2022

### Mount Kenya University · Computer Science

PROJECTS

## Backtest API Trading infrastructure

Private, described architecturally

Backtest as a service: the boring parts, built so the same strategy is never computed twice.

5×3 pairs × timeframes · 60/min rate limit · hash cache key

Private repo, architecture only

Python · FastAPI · TimescaleDB · Redis · Docker

## Signly Forex signal verification

Maintained, not live

Verify Telegram trading signal channels before trusting them. Built and shipped, and maintained; not live, with the public site offline.

5 iterations · 11 mo to product · 5 what-if exits

Maintained, not live

Python · FastAPI · Polygon · PostgreSQL

## Rig Semantic codebase graph over MCP

npm v0.3.0 · Show HN

npx rig-constellation gives a coding agent a map of the codebase instead of grep. Local-first, no key, nothing uploaded.

~100 commits, 4 days · 6 index stages · 3 agents wired

npm: rig-constellation · Source

TypeScript · MCP · Node · SQLite

---

Numbers as of 2026-11-03. No live P&L is claimed.