Opeoluwa Osho

AI & Backend Engineer — Go · Python · TypeScript · Agentic Systems · Event-Driven Architecture

Morristown, NJ 973-671-5465 opeoluwa.osho.star@gmail.com opeoluwa-osho
opeoluwaosho.vercel.app adam-eques

Profile

AI and backend engineer with 7+ years of production backend experience across Go, Python, and TypeScript, now building AI systems end to end — an AI advertising platform I architected from scratch on Google Cloud (exactly-once event processing, command queues, automated QA gates), sustaining 20–30k ads/day; RAG document intelligence for finance teams; and Stable Diffusion inference serving 2–4M monthly visits. I bridge the gap between LLM APIs and real business systems: agentic orchestration, retrieval pipelines, and event-driven services that stay correct under load, retries, and partial failure.

Core Skills

Agentic AI & Multi-Agent Systems — LangGraph · CrewAI · MCP (Model Context Protocol) · event-driven multi-service AI architectures · autonomous agent pipeline design · tool-use and agent-action patterns · QA gating, exactly-once event processing, retries · agent orchestration at scale (Cloud Pub/Sub, Cloud Tasks)
LLM Evaluation & Observability — Automated QA gating before spend · eval harnesses | and grounding checks · production eval loops (live benchmarking → prompt selection) · | citation/groundedness verification · LangSmith
Workflow Automation — n8n (incl. self-hosting & AI nodes) · Make.com · Zapier · LLM-to-SaaS integration
Backend & Full-Stack — Go / Golang · Python · TypeScript / JavaScript · Fastify · Prisma · Server-Sent Events (SSE) · microservices & event-driven architecture · RESTful & WebSocket APIs · React · PostgreSQL · Redis
AI Infrastructure & Serving — GPU inference orchestration (RunPod, AWS SageMaker) · vector databases (Pinecone, pgvector) · dynamic request batching · concurrent job queues & scheduling · inference latency optimization
Infrastructure & Delivery — Docker · Kubernetes · Linux · Terraform · CI/CD (GitLab, CircleCI, Jenkins) · AWS · Google Cloud Platform (Cloud Run, Pub/Sub, Cloud Tasks, Secret Manager) · mTLS / secure communication

Projects

Nova — AI Advertising Automation Platform (20,000–30,000 ads/day) , *Python (FastAPI) services · TypeScript · Node.js (Fastify) · PostgreSQL (Prisma) · React 19 · Google Cloud (Pub/Sub, Cloud Tasks, Cloud Run) · Google Gemini (text, vision, image) · Meta Marketing API*

- Architected from scratch a **production AI ad-automation pipeline on Google Cloud** — seven specialized services (page generation, content, image generation, QA gating, CMS publish, ad launch, budget allocation) coordinated by an event-driven orchestrator I authored solo: **Cloud Tasks** command queues out, **Pub/Sub** events back, sustaining **20,000–30,000 ads/day**.
- Engineered reliability over at-least-once messaging: message-ID deduplication for **exactly-once event handling** and a **monotonic status state machine**, so duplicate or out-of-order events can never corrupt campaign state; failed stages are retryable per-stage from the console.
- Integrated Google Gemini across the pipeline — LLM-written marketing copy, AI image generation with automated visual-QA checks, and multimodal vision scoring that ranks creative styles per product to steer prompt selection: a continuously running production evaluation loop, not an offline benchmark.
- Defined the typed cross-service contract (action names, Pub/Sub topics, payload schemas) implemented by the teammate-built content service; automated QA gates check every page and creative before any ad spend.
- Built the **React 19** operator console (24 routes, role-based access) with **10+ live Server-Sent-Events streams** — bulk generation with 10+ prompt controls, ad-set management with live Meta sync, budget-allocation runs, and AI creative monitoring; took the whole system from a 2025 n8n prototype to production in ~6 months (backend + frontend).

Finance Document Intelligence — RAG Pipeline for Finance Teams,
Python (asyncio) · RAG (Retrieval-Augmented Generation) · pgvector (PostgreSQL) · Claude (Anthropic API) · streaming API

- Built a **RAG pipeline** end-to-end for a finance client — ingesting thousands of pages of financial documents (earnings reports, regulatory filings, contracts), chunking and vector-embedding them into a semantic index with sub-second retrieval across the entire corpus — **choosing pgvector inside PostgreSQL over a managed vector DB so metadata and vectors live in one operationally simple database**.
- Designed the **retrieval and Q&A layer** so a 5-person finance team asks natural-language questions and gets answers grounded in source context with citations — anchoring every response to retrieved passages rather than model weights, sharply reducing hallucination risk.
- Built the **asyncio streaming response API** connecting vector store, retrieval layer, LLM and chat interface; tuned chunk size, overlap and retrieval parameters — cutting document-lookup work that took the team hours of manual cross-referencing down to seconds.

Yodayo — Stable Diffusion performance optimization, Go backend,
Go (Golang) · Stable Diffusion · Illustrious-XL · Flux · RunPod · AWS SageMaker · GLM-4 · DeepSeek V3 · Claude

- Optimized the **Stable Diffusion generation path** for a platform serving **2–4M monthly visits** (peaked at 12M/month) — dynamic request batching (one GPU pass → multiple images), concurrent dispatch and worker pooling to keep RunPod / AWS SageMaker workers saturated, and warm model pools to cut cold-start latency; more images per GPU-hour at lower per-image cost.
- Built the Go generation services (intake, queueing, dispatch, result handling) using **goroutines and channels** to coordinate thousands of in-flight jobs — a backend also powering the **Moescape AI platform** on the same architecture.
- Profiled and tuned the Go hot path with **pprof** — eliminated allocations, lock contention and serialization overhead; added backpressure and retries so traffic spikes degrade gracefully.

Professional Experience

AI Engineer, *Self-Employed* Aug 2024 – Present | Remote

- Designed **Nova** from scratch and built its core — AI ad-automation platform sustaining **20–30k ads/day** (TypeScript, Python/FastAPI services, Google Cloud, Google Gemini, Meta Ads API); took it from an n8n prototype to a **production event-driven backend** in ~6 months, collaborating with the content-service team through a typed integration contract I defined.
- Delivered a **Finance Document Intelligence RAG pipeline** — end-to-end document ingestion, semantic retrieval and grounded Q&A for a 5-person finance team.
- Optimized **Stable Diffusion inference** on Yodayo's Go backend (2–4M monthly visits) — dynamic batching, GPU scheduling (RunPod / SageMaker), pprof tuning; raised throughput and cut latency at peak load.
- Built **n8n, Make.com and Zapier** workflows for multiple clients — LLM-powered lead enrichment, document routing and notification pipelines, eliminating hours of manual data-handling per week.

Backend Engineer, *Infiniteproxies* Aug 2022 – Aug 2024 | Remote

- Redesigned the billing **PostgreSQL** schema and query paths across the company's **Go, Python, and TypeScript backend services**, cutting p95 read latency from ~410ms to ~290ms.
- Added a **Redis** LRU cache for session/usage data, reducing average DB round-trips per request by roughly a third.
- Restructured **Docker** image layers and build caching, cutting CI image builds from ~9 min to ~4.5 min.

Backend Developer (Contract), *FedEx* Sep 2020 – Aug 2022 | Remote

- Built **Go, Python, and TypeScript microservices** for a real-time shipment-tracking system serving ~12K monthly active users, with configurable **WebSocket** APIs for live location/status.
- Implemented Go-based alerting that reduced missed/duplicate notifications to under 1% in production.
- Migrated builds to **Jenkins** pipelines, cutting deploy prep from ~40 min to under 10.

Backend Developer, *Feather* Oct 2018 – Aug 2020 | Remote

- Built scalable **Go and Python microservices** on **Kubernetes** and **Docker** with **PostgreSQL**, serving a user base that grew past ~100K active users.
- Introduced **Redis** caching on hot endpoints, roughly halving p95 response time under peak traffic.
- Secured all service-to-service traffic with **mTLS** end-to-end encryption; maintained consistent sprint delivery.

Education

B.Sc., Computer Science, *University of Lagos* Sep 2012 – Jan 2017 | Lagos, Nigeria