Last updated in Mar 2026

Nyasha Hama

Full-Stack & Systems Engineer | Go, Java, C++, C#, Python, Rust, PostgreSQL | Distributed Systems & Query Engines

Pretoria, South Africa | nyashaahama@gmail.com | portfolio-topaz-one-58.vercel.app
github.com/nyashahama | linkedin.com/in/www.linkedin.com/in/nyasha-hama-5b1312229

Summary

Backend and systems-focused engineer who builds with intention — from production-grade REST APIs in Go to a handwritten query engine in C++ exploring how databases actually work under the hood.

Specialized in Go and C++ with deep interest in distributed systems, query engines, storage internals, and high-performance backend architecture — the kind of engineer who reaches for pgx and sqlc over an ORM because the abstraction cost matters.

Solved 200+ algorithmic problems (LeetCode medium & hard), not as a ritual but as a way of sharpening the performance-first instincts that show up in production — in schema design, indexing strategy, and execution path choices.

Driven by the intersection of correctness and performance — the kind of problems where getting the data model right, the query plan efficient, and the system observable are what separate good engineering from fast engineering.

Skills

Languages: Go, C++, TypeScript, JavaScript, SQL

Backend & APIs: Gin, Node.js, REST APIs, JWT authentication, Middleware design

Databases: PostgreSQL, query optimization, indexing strategies, schema design

Systems & Infrastructure: Docker, Linux (Ubuntu), AWS fundamentals, concurrency, memory management

Computer Science Foundations: Data Structures, Algorithms, Graphs, Trees, Concurrency, System Design

Experience

Software Engineer, Independent Software Engineer – Remote — Jan 2023 – present

- Designed and built production-ready REST APIs using Go (Gin) and PostgreSQL.
- Implemented JWT-based authentication, role-based authorization, and secure session handling.
- Structured scalable backend templates for rapid development of real-world applications.
- Applied indexing and query optimization techniques to improve database performance.
- Containerized applications using Docker for reproducible development and deployment.

Projects

Healthcare Access Connector — Dec 2025 – Feb 2026

Full-stack healthcare platform improving access to clinics, telemedicine, and appointment scheduling for underserved communities.

- Designed RESTful APIs in Go powering patient-provider matching workflows.
- Implemented secure JWT authentication and role-based access control.
- Integrated PostgreSQL with optimized schema design for appointment and clinic data.
- Used Redis for caching high-frequency queries to improve response latency.
- Repository: https://github.com/nyashahama/healthcare-access-connector-backend

Asymmetric Risk Mapper Feb 2026 – Mar 2026

Risk analysis tool that identifies asymmetric business threats — highlighting high-impact risks while filtering low-leverage noise.

- Built backend logic in Go with TypeScript-based analysis layer.
- Designed data models to quantify and rank asymmetric risk exposure.
- Focused on clarity of signal vs noise in decision-making systems.
- Repository: https://github.com/nyashahama/asymmetric-risk-mapper
- Live: https://asymmetric-risk-mapper.vercel.app/

Burnout Predictor (In Progress) Mar 2026 – present

Predictive burnout detection tool analyzing workload patterns to give early warnings before burnout occurs.

- Built full-stack system using Next.js, TypeScript, Go, and PostgreSQL.
- Designed backend services to analyze sleep patterns, work schedules, and calendar load.
- Implemented early-warning logic to estimate burnout risk up to 14 days in advance.
- Focused on turning behavioral signals into actionable risk insights.
- Repository: https://github.com/nyashahama/burnout-predictor
- Live: https://burnout-predictor-ten.vercel.app/

AI Life CFO (In Progress) Mar 2026 – present

Predictive cash flow intelligence platform helping high-earning professionals understand long-term financial sustainability.

- Built marketing and product interface using Next.js, TypeScript, and Tailwind.
- Designed concept for AI-driven cash flow forecasting and lifestyle sustainability analysis.
- Implemented modern UI patterns including scroll-snap presentation layout and custom cursor interactions.
- Focused on turning financial data into forward-looking decision intelligence.
- Repository: https://github.com/nyashahama/AI-Life-CFO
- Live: https://ai-life-cfo.vercel.app

Music Awards System June 2024 – Dec 2024

Scalable full-stack awards management system handling nominations, voting, analytics, and role-based authentication.

- Built backend in Go using Gin and PostgreSQL.
- Implemented role-based permissions for admins, nominees, and voters.
- Dockerized services for consistent local and production environments.
- Integrated CI/CD pipelines using GitHub Actions.
- Repository: https://github.com/nyashahama/music_awards_server

Go Scalable Auth Boilerplate Dec 2025 – Dec 2025

Production-ready authentication backend template built for scalability and performance.

- Implemented pgx + sqlc (no ORM) for type-safe, high-performance SQL access.
- Integrated Redis caching and NATS messaging for scalable architecture.
- Exposed Prometheus metrics for observability and performance monitoring.
- Designed clean layered architecture for extensibility in SaaS systems.
- Repository: https://github.com/nyashahama/go-scalable-auth-boilerplate

E-commerce Search Optimization Jan 2025 – Mar 2025

Java backend system optimizing product search using advanced data structures.

- Implemented graph-based search traversal and BST-backed indexing strategies.
- Applied concurrent maps to improve performance under load.
- Designed relational schemas in PostgreSQL for optimized query execution.
- Repository: https://github.com/nyashahama/Optimizing-Search-Algorithms-in-E-commerce-Platforms-backend

Wedding Planning System Jan 2024 – Dec 2024

Collaborative full-stack university project built with React and Node.js.

- Designed RESTful APIs using Express with structured routing and validation.
- Integrated PostgreSQL for persistent storage.
- Applied clean architecture principles for separation of concerns.
- Repository: https://github.com/nyashahama/Team-34-Project

Mini Query Engine (In Progress) Jan 2026 – present

Experimental C++ project exploring database internals and query execution.

- Designing table abstractions with rows, columns, and schema metadata.
- Implementing execution operators (scan, filter, projection).
- Studying memory layout, execution pipelines, and storage formats.
- Inspired by systems like PostgreSQL and DuckDB.

Education

University Johannesburg, Bachelor's Degree in Computer Science – South Africa Feb 2022 – Dec 2024

- Strong focus on algorithms, systems programming, and backend development.
- Final-year project centered around scalable application architecture.

Achievements

- Solved 200+ algorithmic problems (LeetCode medium & hard).
- Built multiple production-grade backend systems from scratch.
- Transitioned from pure algorithmic problem solving to real-world systems engineering.
- Strong Linux-based development workflow on Ubuntu (Dell XPS 13).

Interests

Distributed systems and database internals — particularly how systems like PostgreSQL and DuckDB handle storage, execution, and consistency at scale.

Query engines and storage engines — actively building a mini query engine in C++ to explore execution pipelines, memory layout, and storage formats from first principles.

High-performance backend services — obsessed with low-latency design, from pgx/sqlc over ORMs to Redis caching and Prometheus-observable architectures.

Building developer tools — enjoy crafting sharp, Unix-philosophy CLI utilities in C++; drawn to tools like fzf that do one thing exceptionally well.

Asymmetric thinking in systems design — applying risk-asymmetry reasoning (from the Asymmetric Risk Mapper project) to backend architecture decisions.

Healthcare infrastructure and access equity — motivated by the real-world impact of the Healthcare Access Connector; interested in how well-engineered systems can solve underserved community problems.

Systems programming and memory models — drawn to C++ and Rust for their explicitness around ownership, layout, and performance.