

SAMUEL KAOMA

Software Engineer · Backend & Distributed Systems · Full-Stack Product Delivery

Lusaka, Zambia | 0762112817 | samuelkaomaa@gmail.com | github.com/Samuelkaoma

PROFESSIONAL SUMMARY

Software engineer who builds systems in which correctness is a requirement rather than an aspiration. Designed and delivered a **sovereign real-time gross settlement core** with threshold-signature authorisation and a Merkle transparency log, a **calibrated forensic evidence-fusion engine** with likelihood ratios that cannot be reported uncalibrated, and a **production Go/PostgreSQL B2B platform** live on Railway. Fluent across **Go, Java 21, Python, TypeScript, C# and C**, with a bias toward event-sourced determinism, hexagonal boundaries enforced by architecture tests, and mutation probes that verify the test suite itself. Roughly **1,600 automated tests** and 128 mutation probes across the portfolio below. Currently building a multi-tenant SaaS platform for government, parastatal, and higher-learning institutions at AVEC Technologies.

EXPERIENCE

Software Engineer

March 2026 – Present

AVEC Technologies | Lusaka, Zambia

- Leading engineering on a **universal multi-tenant SaaS platform** for Zambian parastatal bodies, government agencies, and higher-learning institutions — middleware-driven tenant scoping, role-and-permission RBAC with per-user overrides, audit logging, and workflow automation.
- Architected and shipped **IntelliHealth**, a multi-tenant clinical, pharmacy, and revenue platform on a Go/Echo/PostgreSQL service layer with a React 19 operator front end.
- Own end-to-end delivery: domain modelling, API contracts, schema migrations, front-end implementation, and production deployment — optimising for low-bandwidth institutional users.

ICT Intern — e-Government Division

Nov 2025 – Feb 2026

Smart Zambia Institute (Office of the President) | Lusaka, Zambia

- Attached to the Office Equipment & Maintenance Services Unit at national headquarters, with direct exposure to **enterprise-scale government ICT infrastructure**.
- Rotated through Systems Development, Networking, and Planning — building working fluency in ICT governance, structured support workflows, and network operations.
- Handled confidential government documents and sensitive technical data under full regulatory compliance.

SELECTED ENGINEERING WORK

SKACE — Sovereign Real-Time Gross Settlement Platform

Java 21 · C# / .NET 8 · React · TypeScript · C (PKCS#11) · PostgreSQL · ISO 20022 · mTLS

- Built a national payment-system settlement core around the Bangladesh Bank failure mode: above a value threshold, funds move only on **t-of-n distinct authoriser signatures over that specific transfer**, so a single compromised endpoint yields one signature and settles nothing.
- Implemented the core as a **single-writer deterministic state machine** — no wall-clock reads, no floating-point money, no order-dependent map iteration — making event-sourced replay byte-reproducible; all three prohibitions are enforced by architecture tests, not convention.
- Delivered an ISO 20022 participant gateway (mTLS, idempotency, signature verification), a multi-tenant React bank portal with **in-browser Merkle inclusion and consistency proofs**, and a C#/WPF central-bank operator console with two-person emergency suspension.
- **691 tests and 128 mutation probes** across 71 delivered features — each probe edits one production line and requires the suite to fail; surviving probes exposed an entirely untested browser session layer and a gridlock-resolution path that bypassed the settlement check.

VIFLAP — Calibrated Multi-Evidence Forensic Linkage Engine

Python · NumPy · SciPy · Hypothesis · doctoral research artefact

- Implemented a full speaker-evidence stack from first principles — **GMM-UBM by EM, i-vector total-variability extraction, LDA/WCCN, and two-covariance PLDA** with the exact closed-form log-likelihood ratio — against NumPy and SciPy only, with no deep-learning framework.
- Built the signal-processing front end (MFCC/LFCC, VAD, LPC formant estimation, YIN pitch tracking, jitter/shimmer/HNR), an analysis-by-synthesis **CELP narrowband channel model**, and an LFCC/GMM spoofing countermeasure with an explicit out-of-domain indicator.

- Fused acoustic, behavioural, and transactional streams with **dependence modelling** (per-missingness logistic, Gaussian latent with exact marginalisation, Gaussian copula), measuring overstatement against the naive-summation baseline on every comparison.
- Encoded three guarantees in the **type system**: absence carries its reason instead of collapsing to LR = 1, a likelihood ratio cannot be constructed without an explicit justified prior, and uncalibrated scores are unreportable. **575 tests** including property-based and import-graph architecture tests; a build-time check fails if the vocabulary of identification appears in any emittable string.

Arcus Investments Platform — Production B2B Commercial Suite (live)

[Go](#) · [Echo](#) · [GORM](#) · [PostgreSQL](#) · [React 19](#) · [TypeScript](#) · [Vite](#) · [Docker](#) · [Railway CI/CD](#)

- Shipped and operate a two-product platform on one identity and permission system: a training and admissions pipeline, and a full commercial CRM — **leads to deals to quotations to contracts to payments**, with document versioning, contract signing, an aged debtor book, and value-based approval controls.
- Extended it through the buy side — sourcing, purchase orders, goods receipt, **landed-cost allocation and per-deal margin** — over a stock ledger with transaction-scoped, self-rolling-back database tests.
- Secured sessions with JWT access tokens plus **rotating refresh tokens in httpOnly cookies**, pluggable S3-compatible storage, and per-route permission enforcement documented route by route.
- Diagnosed and documented a silent-CI class of failure: 59 of 118 tests skipped without DATABASE_URL while every package still reported **ok** and exited 0 — precisely the approvals, stock-ledger, landed-cost and payables tests. Now enforced at 136 tests, zero skips.

SKACE Architect — Regulatory Compliance & Generative Design Engine

[Python](#) · [Google OR-Tools CP-SAT](#) · [Shapely](#) · [Decimal exactness](#) · [doctoral research artefact](#)

- Built a compliance engine that turns Zambian building regulation into executable rules and returns a **citation-carrying, byte-reproducible verdict** — content-addressed value references, canonical JSON, and an immutable ruleset binding.
- Made provenance a type invariant: a GroundedValue cannot be constructed without a ValueOrigin, so **no verdict can rest on an unscaled or unsourced measurement**, and a customary boundary cannot support a violation asserted inside its own positional error.
- Compiled applicable rules plus a household brief into CP-SAT constraints that each carry the rule that produced them, then **re-judged every generated layout with the compliance engine** — which caught the solver placing buildings on the very setback they were generated to satisfy.
- Designed the LLM-orchestration grounding layer so **the model cannot originate a number**: tools return registered value references, the facade refuses raw numeric arguments, and a validator rejects any draft containing a numeral the session never obtained. **267 tests**, including architecture tests that fail the build if domain code ever reaches a language model.

IntelliHealth — Multi-Tenant Clinical, Pharmacy & Revenue Platform

[Go](#) · [Echo](#) · [PostgreSQL](#) · [JWT](#) · [React 19](#) · [TypeScript](#) · [Vite](#) · [Tailwind CSS](#)

- Delivered a tenant-scoped hospital operations backend covering **patient records, visit queues with triage priority, clinical diagnosis capture, ICD-10 and CPT coding lookups, and WHO benchmarking**.
- Built the pharmacy and revenue side: stock receipt and adjustment ledgers, dispensation history, low-stock and expiry alerting, FDA drug lookup, **drug-interaction checking**, and invoice finalisation into a billing cycle.
- Implemented the ControlHub tenant-onboarding plane — provisioning, client-admin creation, document submission with timeline tracking — over middleware-enforced tenant scoping, RBAC with per-user permission overrides, and audit logging on every state change.

Z-SIMP — National Infrastructure Monitoring Platform

[Python](#) · [FastAPI](#) · [React](#) · [Vite](#) · [PostgreSQL](#) · [InfluxDB](#) · [scikit-learn](#) · [Docker](#) · [Nginx](#)

- Architected a sensor-driven monitoring platform with **dual honest data modes** — real USGS public telemetry with a 14-day startup backfill, or a labelled Zambian disaster simulator — and a provenance banner that makes it impossible to present synthetic data as real.
- Layered detection deliberately: **time-based rolling IQR baselines and domain rules stay primary and interpretable**, with IsolationForest activating only past 200 baseline points and never overriding them; severity escalates advisory to warning to critical on consecutive breaches.
- Built the operational plane — incident command with acknowledge, assign, escalate and immutable approvals; province-scoped responsibility chains; connectivity-state tracking; an offline-first ingest contract with duplicate suppression and dead-letter capture; and a no-login public status portal.

SmartFarmer SKACE — Verifiable Agricultural Credit Platform

[Go](#) · [Echo](#) · [PostgreSQL Row-Level Security](#) · [React](#) · [Vite](#) · [Expo](#) · [Python](#) · [XGBoost](#)

- Built four cooperating services in one repository — a Go/Echo API, a React dashboard, an Expo field app, and a Python/XGBoost yield service — that turn everyday farm activity into evidence a lender can trust.
- Verified farm claims against independent, hard-to-fake sources — **Sentinel-2 NDVI, the ZNSDI cadastral geoportal, and GPS geofencing** — and emitted a cryptographically signed, QR-verifiable lender track record with tamper detection.
- Enforced tenant isolation at two layers: request-scoped middleware over **PostgreSQL Row-Level Security that fails closed**, with Go, Vitest, and Python suites green.

Survey & ODK Paradata Platform — Research-Grade Anomaly Detection

Python · FastAPI · Pandas · NumPy · React · Vite · pytest

- Engineered a four-layer pipeline — parsing, **40 engineered features**, validation, and anomaly scoring — over mobile survey paradata, with an interactive analytics dashboard.
- Evaluated honestly against ground truth at **76.1% F1**, with end-to-end threshold assertions in the pytest suite and misclassifications reported rather than tuned away.

TECHNICAL SKILLS

Languages	Go, Java 21, Python, TypeScript, JavaScript, C#, C, SQL
Backend & APIs	Echo, FastAPI, Node.js, Express, NestJS, GORM, REST & ISO 20022 message design
Frontend	React 19, Next.js, Vite, TypeScript, Tailwind CSS, Radix UI, Framer Motion
Data & Storage	PostgreSQL (incl. Row-Level Security), MySQL, InfluxDB, Supabase, Prisma, Drizzle ORM, Alembic
Architecture	Event sourcing, hexagonal / ports-and-adapters, multi-tenancy, deterministic state machines, monorepos
Security	mTLS, PKCS#11 / HSM, threshold signatures, Merkle transparency logs, JWT + rotating refresh, RBAC, audit chains
ML & Numerics	scikit-learn, XGBoost, NumPy, SciPy, Pandas, GMM / i-vector / PLDA, calibration, CP-SAT optimisation
Verification	Mutation probes, property-based testing (Hypothesis), architecture tests, pytest, Vitest, Go test, JUnit
Infrastructure	Docker, Docker Compose, Nginx, Railway, GitHub Actions, Linux, Git
Networking	TCP/IP, OSI model, routing & switching, CCNA

EDUCATION

Bachelor of Science in Computer Science | University of Zambia | Expected Graduation: 2026

Selected Academic Results — University of Zambia Transcript

Course Code & Title	Grade
1st Year — Foundation Sciences	
PHY 1010 · Introductory Physics	B
CHE 1000 · Introductory Chemistry	B+
MAT 1100 · Foundation Mathematics	B+
BIO 1400 · Cellular, Molecular Biology & Genetics	B+
2nd Year — Core Computer Science	
CSC 2901 · Discrete Structures	B+
CSC 2101 · Computer Systems	B+
CSC 2111 · Computer Architecture	A+
CSC 2702 · Database & Information Management System	C+
CSC 2202 · Operating Systems	B+
CSC 2000 · Computer Programming	A
CSC 2912 · Numerical Analysis	A

Course Code & Title	Grade
3rd Year — Advanced Systems & Software	
CSC 3600 · Software Engineering	B
CSC 3301 · Programming Languages Design & Impl.	B+
CSC 3801 · Data Communication & Networks	B+
CSC 3612 · IT Project Management	B
CSC 3712 · Advanced Databases	B+
CSC 3011 · Algorithm and Complexity	B+
CSC 3402 · Fundamentals of Artificial Intelligence	B

Achieved A+ in Computer Architecture — top grade. Three A/A+ results and a majority of B+ across 18 modules spanning three years of study.

CERTIFICATIONS

Cisco CCNA	Cisco Networking Academy
Cisco IT Essentials	Cisco Networking Academy
Cisco Linux Program	Cisco Networking Academy