MOHAMED ALI

Software Engineer & Founder

Architecture that connects system reliability to product momentum.

Backend architecture, mobile systems, and product engineering for teams that need clear decisions, resilient delivery, and measurable operating leverage.

POSITIONING

Backend architecture / Mobile systems / Product engineering

mohakavi.dev

Technical depth is only useful when it changes the business outcome.

I work from the constraint backward: revenue path, reliability risk, operating cost, and delivery speed determine the architecture - not fashion or framework preference.

01

Product leverage

Find the workflow where better engineering changes conversion, retention, or operating capacity.

02

System leverage

Remove race conditions, hidden latency, brittle integrations, and unclear ownership boundaries.

03

Decision leverage

Instrument the product so teams can explain activation, failure modes, and revenue movement.

BEST-FIT ENGAGEMENTS

- A product is reaching scale and reliability problems are becoming business problems.
- A mobile or backend workflow needs a stronger architecture before the next growth phase.
- A founder needs senior technical judgment without adding process theatre.
- A team needs a focused architecture audit, rescue plan, or delivery path.

ENGAGEMENT MODES

Architecture intensive

Constraint map, risk register, target architecture, and sequenced implementation plan.

Build partnership

Hands-on delivery across backend, mobile, web, infrastructure, and instrumentation.

Technical due diligence

A direct assessment of scalability, reliability, maintainability, and delivery risk.

WHAT THE CLIENT RECEIVES

DECISION MAP
Constraints, risks, ownership, and tradeoffs made explicit.

SEQUENCED PATH
A practical build order tied to business and reliability outcomes.

OPERABLE HANDOFF
Instrumentation, failure paths, and operating knowledge included.

02 / CAPABILITY MAP

Senior judgment across the critical product boundaries.

The strongest value appears at the interfaces: mobile to API, API to data, models to queues, and architecture to business operations.

01

Mobile architecture

Flutter / React Native / Swift

- Offline-capable workflows
- Deterministic sync and reconciliation
- State management and performance
- Reliable release paths

02

Scalable backend

Go / Node.js / Spring Boot / FastAPI

- Concurrency-safe service design
- Idempotent APIs and event flows
- Queue and webhook reliability
- SQL and NoSQL data models

03

Modern web

Next.js / React / TypeScript

- Product-grade frontend systems
- Conversion-aware workflows
- Accessible responsive interfaces
- Instrumentation across funnels

04

Infrastructure & AI

Docker / Redis / Ollama / Automation

- Containerized delivery
- Queue-aware model orchestration
- Streaming and backpressure
- Observability and operational tooling

ARCHITECTURE STANDARD

STATE INTEGRITY
Explicit ownership and deterministic transitions.

FAILURE-AWARE
Retries, idempotency, recovery, and degraded modes.

MEASURABLE
Operational and product signals designed into the system.

03 / SELECTED PROOF

Founder products and confidential engineering work, separated clearly.

Named products below are founder-owned. Client and company work is described by system type to preserve confidentiality.

FOUNDER PRODUCT

Oraahyo

Somali knowledge platform

Designed and built the bilingual content model, calm reader, search, accounts, and reading progress end to end.

25+ book summaries / 90+ translated quotes / Web live

FOUNDER PRODUCT

Dhiig-Baaq

Emergency blood-response network

Built donor onboarding, hospital directory, urgent request matching, and network-resilient response flows for Mogadishu.

340+ lives saved / Banadir network / Live platform

CONFIDENTIAL ENGAGEMENT

Vehicle marketplace

Mobile + operations architecture

Introduced deterministic sync, idempotent moderation updates, conflict handling, and cross-platform analytics instrumentation.

Lower duplicate-report volume / Faster moderation / Revenue attribution

CONFIDENTIAL ENGAGEMENT

LLM orchestration

Go-based multi-model infrastructure

Implemented queue-aware routing, warm workers, streaming backpressure, context cancellation, and Redis coordination.

Substantial P95 latency reduction / Stable burst behavior

04 / DELIVERY SYSTEM

Clarity before code. Evidence before confidence.

Every engagement is structured to surface constraints early, reduce avoidable rework, and leave the team with an operable system rather than a fragile handoff.

01	**Frame**	Define the user, business outcome, constraint, and failure cost.
02	**Model**	Map system boundaries, data ownership, state transitions, and risk.
03	**Build**	Ship the smallest reliable vertical slice with explicit tradeoffs.
04	**Measure**	Instrument reliability, product behavior, and operating signals.
05	**Harden**	Test failure paths, document operations, and sequence the next scale step.

ARCHITECTURE INTAKE

Bring the constraint. Leave with the highest-leverage technical path.

Focused consultation for architecture, delivery risk, scale readiness, or product-system alignment.

START CONVERSATION