

Mohamed Elamin.

*engineering practice in public — software, agents,
mobile, embedded.*

// THE SEAMS STAY VISIBLE.

what i do.

a single practice across software, agents, mobile, and embedded — under one name, in public.

// SINCE

2017 → now

// CURRENT ROLE

senior AI engineer • agent
systems
KALVAD • 11/2021 → now

// BASED

dubai • uae

// CONTACT

mohamed@fifty.dev
+971 55 650 7945

// THE PRACTICE

i ship software, automations, agents, mobile apps, and embedded systems — under a single name and a single visual language. the work spans personal infrastructure (IGRIS, fifty.dev, the flutter kit) and shipped commercial products for UAE government and enterprise clients at KALVAD.

the through-line is architecture before strokes; deciding what to build, then building it well. agents that plan and verify before they act. mobile platforms that ship to production and stay maintainable years later. infrastructure that makes truth portable across tools.

i prefer specific over perfect, receipts over claims, and the seams visible. the practice is in public — code, briefs, learnings, and failures land in repos and journals where the work can be inspected.

// CAPABILITY DOMAINS

AI ENGINEERING

AGENT DESIGN

LLM ORCHESTRATION

MOBILE • FLUTTER

BACKEND • PYTHON • TS

DESIGN SYSTEMS

CONTENT SYSTEMS

EMBEDDED / HARDWARE

GAMES & INTERACTIVE

9+

// YEARS SHIPPING

20+

// LIVE PRODUCTS

24+

// OPEN-SOURCE RELEASES

the method.

*four principles that hold across every project in the next eight pages.
nothing is invented per-project — the discipline carries.*

// SINCE

2017 → now

// APPLIED ACROSS

IGRIS • the kit • Hadir •
Opaala • Arkada games • the
content pipeline • 8 other
shipped apps

// ORIGIN

distilled from a decade of
shipping commercial software

// VISIBLE IN

every project that follows

// 01 • ARCHITECTURE BEFORE STROKES

decide what to build, then build it well. the first real design problem is what deserves to exist. the routing tree, the brief schema, the kit's package boundaries — all of them are decisions made before any code is written. shipping fast is downstream of deciding clearly.

// 02 • BRIEF-FIRST

every file modification gates on a brief. the discipline is at the front, not after the fact. when there's no brief, there's no work. IGRIS enforces it as a protocol; the content pipeline enforces it as a "not yet" gate; on client work it shows up as plans approved before diffs land.

// 03 • PLAN, ACT, VERIFY

every change passes a steering point. agents (and engineers) work between the gates; humans (and reviewers) decide at the gates. narrow human surface where I'm the only one accountable, wide agent surface where the work actually happens. same loop in IGRIS, same loop on every team I lead.

// 04 • SHOW THE WORK

drafts visible. failures documented. seams left there on purpose. the practice is in public — code, briefs, learnings, and broken things land in repos and journals where they can be inspected. specific over perfect, receipts over claims.

IGRIS.

an engineering OS for AI coding agents — cross-harness work-state handoff, enforcement-as-code, and one persistent brain.

// ROLE

author • architect • sole engineer

// YEAR

2025 → now

// STATE

LIVE • v7.2.0 • MIT

// STACK

TYPESCRIPT

NODE

SQLITE • FTS5

MCP

CLAUDE CODE

CODEx • GEMINI • OPENCODE

// REPO

github.com/fiftynotai/igris-ai

// PROBLEM

AI coding agents make implementation faster without preserving the structure around the work. context resets lose decisions, parallel sessions collide, and prompt-level rules remain suggestions. the problem was not another coding model; it was an operating layer that keeps work state, ownership, standards, and verification intact across tools.

// APPROACH

built IGRIS as a harness-agnostic engineering OS with brief-first executable gates, bounded delivery roles, atomic work claims, and a centralized SQLite + FTS5 brain. typed edges preserve dependencies and decision lineage; MCP projects the brain across harnesses; an operator dashboard exposes projects, work state, cognition, search, and the graph. current work adds a visual architecture canvas grounded in reusable components from the catalog.

// HUMAN SURFACE

- plan approval
- diff review
- retry-failure call
- anything that touches main

// AGENT SURFACE

- planning the work
- writing code against the plan
- running tests • fixing broken tests
- scoped refactors
- brief management • subagent dispatch
- self-recovery on error • plan revision
- drafting commit messages
- memory writes back to the brain

// OUTCOME

// proven work-state handoff across Claude, OpenCode, Codex, and Antigravity

// live remote brain exposes 112 MCP tools across memory, briefs, goals, graph, cognition, schedules, and reusable assets

// executable brief, phase, review, ownership, commit, and secret-scanning gates

// own daily driver — open-source, MIT, and in active development at v7.2.0

fifty agent sdk.

a production-grade reusable agent loop for Python — custom ReAct, JSON-mode tool calls, MCP, and pluggable backends.

// ROLE

author • architect •
maintainer

// YEAR

2026 → now

// STATE

LIVE • v1.5.0 • MIT

// STACK

PYTHON 3.11+

ASYNCIO

REACT

JSON-MODE TOOLS

MCP

SQL • REDIS

// REPO

github.com/fiftnotai/fifty-
agent-sdk

// PROBLEM

the agent loop, parser, safety checks, state handling, and runner kept being rewritten for every project. each implementation solved the same orchestration problem with slightly different assumptions, failure modes, and infrastructure requirements.

// APPROACH

factored the reusable loop into one Python SDK. LLM clients, tools, state stores, audit sinks, and observability hooks sit behind protocols; in-process and MCP tools share one registry; typed events expose every step; iteration caps and per-tool timeouts keep runs bounded. the default needs no database, while SQL and Redis remain optional backends.

// OUTCOME

- // already used in two work projects
- // any OpenAI-compatible chat-completions endpoint through one adapter
- // memory, SQL, and Redis state stores with first-class conversation branching
- // recoverable MCP tool errors, typed event streaming, audit hooks, and bounded execution

fifty flutter kit.

a curated toolkit for Flutter and Dart — design tokens, UI, forms, caching, audio, games, narrative. battle-tested across a decade of real apps.

// ROLE

author • publisher • maintainer

// YEAR

2024 → now

// STATE

LIVE • 17 packages • MIT

// STACK

DART 3

FLUTTER 3.X

MVVM + ACTIONS

GETX

PUB.DEV

// REPO

github.com/fiftynotai/fifty_flutter_kit

// PROBLEM

close to a decade of building Flutter apps left a pile of internal utilities and tribal knowledge — patterns reached for on every project but kept rewriting. the cost was forks, drift, and reinvention. teams downstream had no shared foundation; the same component shipped slightly differently in every app.

// APPROACH

extracted the most-used patterns into a properly packaged, versioned, open-source toolkit — one source per domain, compiled to every consumer. published under verified publisher fifty.dev on pub.dev.

// THE 17 PACKAGES

fifty_tokens design tokens — colors, typography, spacing, motion	fifty_theme token-driven Material ThemeData	fifty_ui FDL v2 components — buttons, cards, inputs
fifty_forms validation, wizards, draft persistence	fifty_cache TTL-based HTTP response caching	fifty_storage secure token storage + preferences
fifty_connectivity network monitoring + DNS/HTTP probing	fifty_socket Phoenix WebSocket — auto-reconnect, heartbeat	fifty_printing_engine ESC/POS over Bluetooth + WiFi
fifty_scroll_sequence scroll-driven image-sequence animation	fifty_audio_engine modular reactive audio	fifty_speech_engine TTS + STT for games / apps
fifty_narrative_engine story + dialog processing	fifty_world_engine Flame-based grid world rendering	fifty_skill_tree interactive skill-tree widget
fifty_achievement_engine condition-based unlocks + tracking	fifty_utils DateTime, Duration, Color extensions, breakpoints	

// OUTCOME

- // 17 packages published + maintained on pub.dev (verified publisher: fifty.dev)
- // powers Hadir, Arkada Studio games, showcase apps — no forks, no drift
- // MIT-licensed — the receipts of a decade of practice, public

Hadir.

white-label mobile attendance platform. real-time geolocation, multi-tenant deployment, in production across multiple clients.

// ROLE

lead mobile engineer •
architecture • delivery

// YEAR

2024 → now

// LIVE TENANTS

MOCA SSA
FYA Hadir

// STACK

FLUTTER • DART

GETX

MVVM + ACTIONS

NODE BACKEND

FIREBASE

MICROSOFT SSO

GEOLOCATION

// REPO

com.kalvad.hadir_moca

// PROBLEM

workforce-management clients needed an attendance app per organization — branded, location-aware, integrated with their SSO. building one app per client meant near-identical codebases drifting apart within a year.

// APPROACH

built Hadir as a white-label platform; one shared mobile codebase, per-tenant theming, per-tenant backend endpoint, per-tenant SSO integration. Flutter with MVVM + Actions Layer over GetX. Node backend with attendance, CMS, and auth services. live as MOCA SSA — Support Service Attendance, serving contracted support-service teams within MOCA entities.

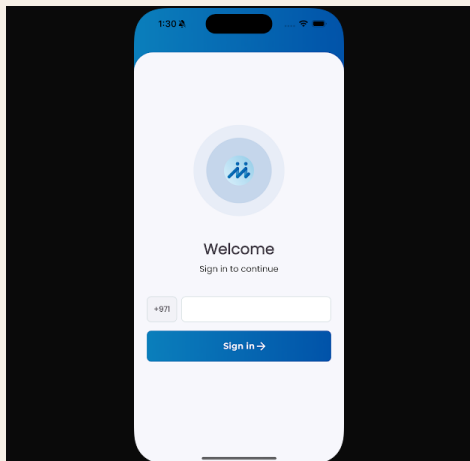
// OUTCOME

// automatic check-in / check-out for field teams

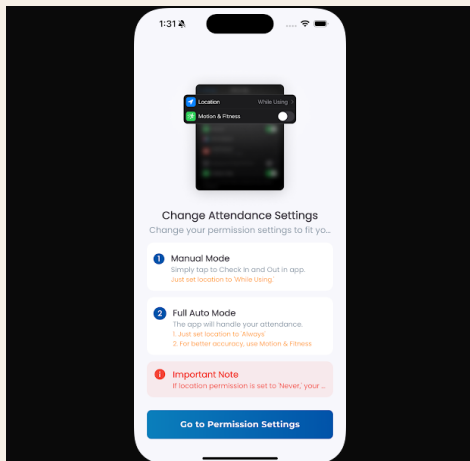
// shift schedules and work assignments

// real-time notifications and announcements

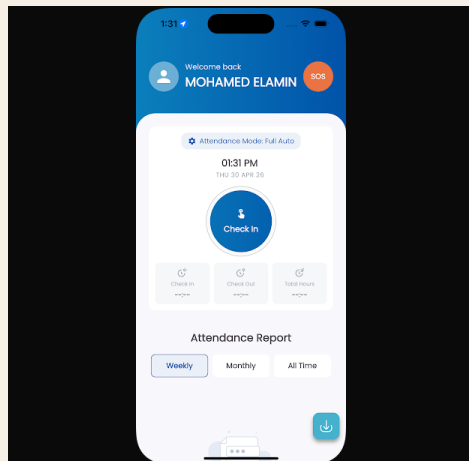
// shared platform, per-tenant branding — no forks, no drift



// MOCA SSA • SIGN-IN



// MOCA SSA • DASHBOARD



// MOCA SSA • CHECK-IN

Opaala.

operator-side admin for the Opaala remote-ordering platform — venues see customer orders come through, process bills, print receipts, every shift.

// ROLE

lead mobile engineer •
delivery • hardware
integration

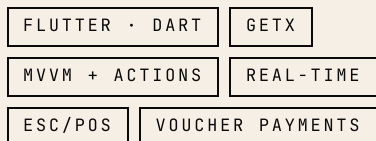
// YEAR

2022 → now

// STATE

LIVE • multi-venue (Opaala
Riverside +)

// STACK



// REPO

com.kalvad.opaala_admin

// PROBLEM

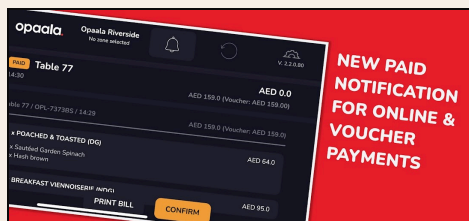
Opaala runs a customer-side remote-ordering platform — diners place orders, venues fulfil them. the operator side needed to be fast, reliable on weak networks, and integrate cleanly with whatever thermal printer each venue already had on the counter.

// APPROACH

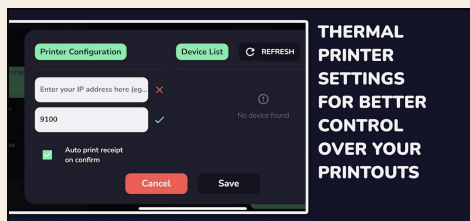
built Opaala Venues as the operator admin; incoming customer orders appear live, with per-table state (waiter, bill, paid, voucher); a "new paid" notification fires the moment an online or voucher payment lands. ESC/POS thermal-printer integration over TCP (port 9100) with configurable per-venue printer IP and auto-print on confirm. multi-venue from day one.

// OUTCOME

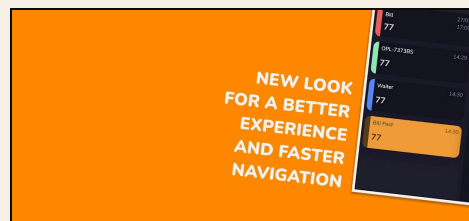
- // operator admin live across multiple Opaala venues
- // thermal-printer integration over TCP with per-venue printer IP + auto-print on confirm
- // real-time paid-notification for online + voucher payments — no manual refresh
- // shared platform with Hadir — same Flutter + GetX + MVVM stack



// OPAALA • TABLE BILLING



// OPAALA • ESC/POS PRINTER



// OPAALA • LIVE STATE

content pipeline.

multi-agent system that decides before it renders. routing tree in front, three agents behind it. this portfolio came out of it.

// ROLE

author • architect • sole engineer

// YEAR

2026 → now

// STATE

WIP • daily driver

// STACK

PYTHON

CLAUDE CODE SUBAGENTS

HIGGS CLI

HEADLESS CHROME

// REPO

~/automation/content

// PROBLEM

generating a post is easy. deciding whether it has earned the right to exist is the harder part. the first attempt was a wall of conditions piped to a single LLM — fast, prolific, off-brand. the second attempt had to be the opposite; a system that says “not yet” before it spends a token.

// APPROACH

three-agent pipeline with strict separation. DECK plans (idea → channel → format → vocabulary budget → beats). WRITER drafts to a locked spec in brand voice. DESIGNER routes to an archetype library and emits the render prompt. in front of all three; a routing registry that decides what earns a post, a brief schema that keeps idea, deliverable, and asset separate, and gates whose main job is to say “not yet.”

// THE PIPELINE

- DECK • plan
- WRITER • copy
- DESIGNER • render

// OUTCOME

- shipped 19 LinkedIn posts in a sustained six-week run; now restarting on a consistency-first cadence
- brand book, CV, LinkedIn brand assets, and this portfolio all rendered through it
- renders dispatch per-asset — higgs CLI for images, headless Chrome for print-family PDFs
- extends to 4 CLI harnesses via igris-brain MCP distribution

// OUTCOME

- // shipped 19 LinkedIn posts in a sustained six-week run; now restarting on a consistency-first cadence
- // brand book, CV, LinkedIn brand assets, and this portfolio all rendered through it
- // renders dispatch per-asset — higgs CLI for images, headless Chrome for print-family PDFs

Erune - Spark Riders 3000.

two mobile companion apps for Arkada Studio board games — the app is the game master; the Spirit of Erune runs the adventure, Iris the spaceship AI runs the mission.

// ROLE

lead mobile engineer • speech
+ narrative engines

// YEAR

2022 → 2024

// STATE

LIVE • both titles published

// STACK

FLUTTER • DART

SPEECH-TO-TEXT

TEXT-TO-SPEECH

NARRATIVE ENGINE

FIFTY KIT

// REPO

com.arkadastudio.erune

// PROBLEM

physical board games with companion apps usually do one thing; keep score. Arkada wanted the app to be the game master — running the adventure end-to-end; tracking state, narrating story beats, voicing NPCs, gating progression on what players actually said out loud.

// APPROACH

built a speech layer (STT + TTS) wired into a narrative engine that owns world state, dialog trees, scene transitions, and reveal logic. Erune — the app is the Spirit of Erune, GMing the adventure and embodying 50+ NPCs. Spark Riders 3000 — the app is Iris, the spaceship's AI, GMing each mission. both apps share the same Flutter engine stack.

// OUTCOME

- // two titles published under Arkada Studio
- // Erune — 50+ NPCs voiced by the Spirit; the app runs the whole adventure
- // speech + narrative + audio engines packaged for reuse in the open-source fifty.dev kit



// ERUNE • 50+ CHARACTERS



// ERUNE • WORLD



// SPARK RIDERS • MISSIONS

also shipped.

the rest of the live work — every entry below is in production right now, public, and verifiable.

LIVE IN PRODUCTION

	MOCA Smart COM.EXARCPLUS.PMO • UAE PMO	// LEAD	internal employee app for the UAE Prime Minister's Office — digital services and automatic location-based attendance on arrival at the office.
	Injaz COM.PMO.INJAZ • UAE PMO	// LEAD	projects and task management system for the UAE Prime Minister's Office — mobile surface for tracking, assignment, and progress.
	Customer Pulse COM.PMO.CUSTOMERPULSE • UAE PMO	// LEAD	citizen-facing app for rating UAE government services and service centers, and submitting surveys provided by government entities.
	EDB Infinity COM.EDB.INFINITY • SUDAN EXPORT DEVELOPMENT BANK	// LEAD	mobile banking for Sudan Export Development Bank — online account opening, statements + balance, internal + BBAN transfers, QR merchant pay, electricity, telco bill pay (Zain · Sudani · MTN).
	Saydak صيدك COM.SAYDAK.COM • STARTING VISION EST	// LEAD	Saudi-based seafood marketplace (Jeddah) — connects fishermen and customers, with peeling, cooking, and delivery services. e-payment via mada (مدى) — Saudi's national payment scheme.
	Al Riyadh جريدة الرياض ID777620825 • YAMAMAH PRESS	// LEAD	official iOS app for Al Riyadh — Saudi Arabia's daily newspaper from Yamamah Press Est.
	fCar ID1564023010 • NABIL YAGHUB IBRAHIM OSMAN	// LEAD	ride-hailing transportation platform — passenger ↔ captain matching, on-screen ride tracking, electronic-wallet payment.
	Rita-Pay ريتاPay SD.RITTAL.CONSUMER • RITTAL • SUDAN FINTECH	// LEAD	consumer mobile payments for Sudan — electricity purchase, telecom bill pay, top-up, card-to-card transfer, voucher generation.

PRIOR VENTURES

	Sukoon @WRITERSCORNERSD • WRITERS CORNER SD • 2019-2020	// CO-FOUNDER	social network for readers and writers — book library by category, story posting with comments and ratings, in-app reading mode, follow-and-feed. Flutter.
--	---	---------------	--

OPEN SOURCE & WEB

	fifty.dev THE WORKBENCH	// AUTHOR	personal practice site + brand book v2 — design tokens, voice, motion system. powers every artifact in this portfolio.
	igris-brain MCP GITHUB.COM/FIFTYNOTAI/IGRIS-AI	// AUTHOR	MCP server bundled with IGRIS — projects one persistent brain across every supported harness.