AHMAD KHIDIR

Berlin, Germany | akhidir2020@gmail.com | linkedin.com/in/ahmad-khidir | github.com/ahmadkhidir

SUMMARY

Software Engineer with 4+ years of experience building scalable backend systems and cross-platform applications across fintech, real estate, health-tech, and AI. Strong focus on system reliability, cloud infrastructure (**AWS**/**GCP**), real-time systems, and AI-powered features. Proven impact in reducing outages by **80%**, scaling platforms to thousands of users, and shipping full-stack products end-to-end across **Flutter**, **Next.js/React**, and **FastAPI/Django/Go** stacks.

WORK EXPERIENCE

SOFTWARE ENGINEER | SAFEPASS | MAY 2026 – PRESENT

- Built a safety-focused navigation and incident-intelligence platform with live human-monitored road-trip tracking, emergency response, panic-triggered silent audio recording, and crowdsourced route-safety markers.
- Developed an event-driven **Flutter** mobile app with **Firebase Auth** (Google/Facebook/Apple + phone OTP), foreground-service background GPS tracking, offline queueing (**hive** + secure storage), and panic-button audio recorded in 30s self-contained chunks with retry-until-confirmed upload.
- Designed a **Hono + TypeScript** backend (23 route modules) with **PostgreSQL (RDS)** for durable data, **DynamoDB** (60s TTL) for ephemeral live GPS state, **Redis (Upstash)** for rate limiting/presence, and native **WebSockets** for live tracking, messaging, and emergency events.
- Implemented unified auth via Firebase Admin SDK token verification + **JWT** (15-min access / 7-day refresh), **RBAC** across 6 roles, and an admin role-upgrade approval workflow with transactional email (**Resend**).
- Built a wallet-based billing system with **Paystack/Flutterwave** webhooks (HMAC-SHA512 verification, atomic debits), pay-per-journey pricing, and org slot-based subscription billing.
- Designed durable trip archival (per-trip summaries + sampled location history for route replay) with admin-only access and NDPR-style account deletion with 14-day cooling-off and anonymization.
- Provisioned **Terraform** IaC (VPC, **ECS Fargate**, RDS Multi-AZ, DynamoDB, **S3 Object Lock**, CloudFront, ACM, Upstash) with **GitHub Actions OIDC** CI/CD and rolling deploys.
- Hardened security: **TLS 1.3**, **AES-256/KMS** encryption, evidence chain-of-custody (hash + GPS metadata + Object Lock), circuit breakers on payments/maps, and a dedicated panic-alert priority path with **SMS** fallback.

SOFTWARE ENGINEER | DE-DUKE GARDEN CARE, LAGOS | MAR 2025 – APR 2026

- Led a real-estate marketplace connecting guests with verified hosts for Commercial (sale/lease) and Shortlet (rental) listings, monetized via two-sided commission.
- Built a **Flutter** mobile app and an async-first **FastAPI** backend deployed as stateless containers on **AWS Fargate** behind an **ALB** with target-tracking auto-scaling and a connection pooler.
- Architected geospatial “near me” search via **PostGIS** and semantic search via **pgvector** embeddings, with cursor-based pagination, read replicas from launch, **Redis** caching, and timeout/circuit-breaker fallbacks to keyword search.
- Implemented real-time three-way support chat (client, host/agency, staff) in **Cloud Firestore** with document security rules + **Firebase** custom claims, alongside Firebase Auth (Google, email/password, phone OTP).
- Engineered payment correctness: **Paystack** checkout, 15-min booking holds with auto-expiry, client idempotency keys, signature-verified webhooks, admin-only escrow release → host/agency wallets → automated **Paystack Transfer** withdrawals with failed-transfer reversals.
- Built type-specific host verification (Owner/Agent/Company/Lawyer/Architect/Surveyor) with manual staff review, and listing moderation that auto-approves professional-host listings.
- Delivered a **Next.js** Admin Web Console with moderation queue, verification review, dispute/refund workflows, chat oversight, commission-rate config, Release Funds queue, and ops/revenue analytics dashboards — all sensitive actions audit-logged.
- Provisioned all AWS infra (networking, Fargate, RDS, Redis, S3+CDN, WAF, Secrets) with **Terraform** and split-path **GitHub Actions** CI/CD (ECR image builds, rolling deploys, functional + performance smoke tests with auto-rollback).
- Authored a **k6** load-testing suite targeting 50,000 concurrent users, 2,000 req/s, and a 5M-listing catalog with a hard zero-double-booking / zero-duplicate-charge gate.
- Applied NDPR-aligned data retention/deletion, **/v1** API versioning, and expand-contract database migrations.

MOBILE APP ENGINEER | PINNAVIEW NETWORKS, IBADAN | MAY 2024 – JAN 2025

- Developed the cross-platform mobile application from the ground up, ensuring a seamless user experience across Android and iOS platforms. Utilized: **Flutter**, **Dart**, **BLoC**, **Dio**.

Ahmad Khidir

- Built a virtual account funding system to automate wallet payments, which significantly increased transaction volumes. Utilized: **REST APIs**, **Payment Gateway Integration**.
- Implemented in-app and push notifications to give users instant transaction visibility.
- Designed and integrated core features for bulk data purchases, cashback rewards, and referral systems.
- Optimized application performance by refining API communication and caching strategies. Utilized: **Hive**, **Repository Pattern**, **Secure Storage**.

BACKEND ENGINEER | POPAT LIMITED, LAGOS | SEPT 2023 – APR 2024

- Restored and stabilized critical production and staging environments, identifying root causes of system failures and reducing outages by **80%**. Utilized: **Linux Server Administration**, **Nginx**, **System Monitoring**.
- Optimized backend service configurations to increase server stability and improve performance across messaging and task-processing systems. Utilized: **SSL/Certbot**, **Celery**, **Redis**.
- Wrote comprehensive API documentation to standardize communication between services and significantly reduce integration errors for the development team. Utilized: **Swagger**, **OpenAPI**.
- Developed a calendar system for event-tracking to drive higher user engagement and platform discovery. Utilized: **Django**, **JavaScript**, **Database Schema Design**.
- Built an email dispatch system and custom templates to facilitate consistent communication. Utilized: **SMTP Services**, **HTML/CSS Email Templating**.

SOFTWARE ENGINEER | LIFECOEL LLC, WASHINGTON DC, USA | FEB 2021 – JUL 2023

- Developed travel update modules for a travel risk management application. Optimized booking and medical test scheduling interfaces to improve user flow. Utilized: **React.js**, **React Native**, **Redux**, **Axios**, **Babel**, **Jest**.
- Built and scaled a career-development SaaS to **1,000+ users**, managing the project through the full development lifecycle. Utilized: **Django**, **React**, **AWS**, **PostgreSQL**, **OAuth**, **Docker**.
- Developed and deployed a cross-platform mobile suite, serving **5,000+ daily users**. Reduced load times through code profiling and state management. Utilized: **Flutter**, **Dart**, **BLoC**, **Provider**, **REST APIs**, **WebSockets**.
- Automated development workflows by implementing **CI/CD** pipelines and optimizing database queries. Reduced deployment times by **50%** and increased system stability and application speed by **40%**. Utilized: **Jenkins**, **Docker**, **Django ORM**, **Redis**, **Nginx**.

PROJECTS

ARK-MASK — AI FACELESS VIDEO GENERATION PLATFORM

- Built a mobile-first platform that turns written stories into faceless AI videos for creators (YouTube/TikTok/IG Reels), with an Obsidian-style project file browser, **MDX** editors, and offline editing via Firestore SDK offline persistence.
- Architected a multi-stage AI pipeline: story → asset extraction → image prompts → reference images → storyboards → audio-included scene videos → cloud-merged final export.
- Engineered a unified AI-provider abstraction (**Google Gemini** image/video models, **BytePlus Ark**) with BYOK keys forwarded in-flight and never stored server-side, plus provider fallback.
- Implemented async generation workers (image, video, cloud-side **FFmpeg** merge) via **Cloud Tasks + Cloud Run**, with **FCM** push completion and **Firestore** real-time listeners driving UI updates.
- Designed **Cloud Firestore** as a unified content + operational data store with security rules and O(1) hashed-platform-key lookup; permanent **GCS** media storage served via 1-hour-TTL presigned URLs.
- Implemented credit-based freemium billing with **Stripe** (Free / Creator $9/mo / Studio $29/mo), atomic Firestore credit transactions, and refunds on provider failure.
- Built an in-app timeline video editor (per-clip trim, cut/fade/dissolve transitions) with server-side merge; no media stored on device.
- Hardened security: bcrypt-hashed platform API keys, hardware-backed secure storage (Keychain/Keystore), scoped GCS path authorization, TLS, GDPR-compliant deletion.
- **Stack:** Flutter/Dart (BLoC, GoRouter, Hive), FastAPI (SQLAlchemy/Alembic, Pydantic v2), GCP Cloud Run/Cloud Tasks/Firestore/GCS, Firebase Auth + FCM, Stripe, Docker Compose + MinIO.

GROUNDWORK — AI DATA ANALYTICS PLATFORM

- Built an AI analytics platform where users upload CSV/Excel/Parquet/SQLite and get AI-narrated, chart-illustrated answers where every number traces to an actual query result.
- Engineered a two-stage grounded-answer loop: model writes SQL → disposable **DuckDB** engine executes → model narrates over facts (never data-in-context), using **Anthropic Claude** via the official Go SDK.
- Architected a single **Go** backend (chi, pgx, goose) owning a durable, resumable run/step DAG orchestrator in **Postgres** (leases, heartbeats, idempotency keys, LISTEN/NOTIFY), streamed live to the UI over SSE.
- Implemented layered verification: determinism cache, static SQL analysis, post-execution reconciliation, and claim-level provenance binding narrative numbers to backing cells.

Ahmad Khidir

- Built an ingest pipeline with type inference (per-type fit-rate confidence badges), PII detection at ingest, multi-file relationship inference with confidence scoring, and mandatory user confirmation.
- Shipped privacy controls: shared payload builder stripping PII before model calls, fail-closed egress audit log, and one-workspace-one-delete cascade.
- Deployed on scale-to-zero infra: **Fly.io** (per-session engine processes), **Neon** Postgres, **Cloudflare R2** Parquet storage.
- Web: **Next.js 16 App Router + React 19 + Tailwind v4** with first-party SVG chart rendering and Auth.js (next-auth v5) JWE cookie auth.
- **Stack:** Next.js 16, React 19, TypeScript, Tailwind v4, Go 1.26, chi, pgx, goose, duckdb-go, anthropic-sdk-go, Fly.io, Neon, Cloudflare R2, Vitest, Playwright.

NEARBY — AI LANDMARK FACTS APP

- Built a **Flutter + Serverpod** monorepo app (iOS + Android) that pings AI-generated facts about landmarks via background proximity detection.
- Implemented background proximity detection with `flutter_background_geolocation` + `geolocator`, landmark matching with on-demand coverage backfill (~1.1km grid cells, 30-day memo, per-device daily caps).
- Integrated **Gemini via Vertex AI** for real-time ~10-fact generation with fail-soft behavior and a server-enforced 50/month free-tier cap; auth via Application Default Credentials (no API keys).
- Implemented 16 features + 10 screens including a motion-rich fact-reveal card (**Rive**), favorites, discovery history, clustered map view, and detection-radius settings.
- Architected a client-server split (Geofence Ingestion, Landmark Matching, Fact Generation, Notification Dispatch, Core API, Map Service, Billing/Entitlement) with **Serverpod**, **PostgreSQL + PostGIS**, **Redis**, and **FCM + APNs** push.
- Shipped freemium billing where entitlement is server-authoritative (verifies Apple/Google purchases and fails closed when the store is unreachable).
- Implemented a design-token system mapped into native **Flutter ThemeData** with a single motion driver, reduced-motion fallbacks, and WCAG AA accessibility (48px touch targets).
- Shipped 237 mobile + 191 API tests green.
- **Stack:** Flutter 3.44, Dart, Riverpod, go_router, Rive, Serverpod, PostgreSQL 16 + PostGIS, Redis, Gemini/Vertex AI, FCM/APNs, Docker.

DRAMATIX — AI DIALOGUE-DRIVEN VIDEO STORY PLATFORM

- Built an AI platform where characters tell stories through multi-voice animated scenes (Flutter mobile + serverless backend).
- Architected a two-stage AI generation pipeline — scene-image generation anchoring visual context, then unified video+audio+dialogue generation — with retry and quality validation.
- Designed a model abstraction layer (**ModelRouter** + adapters) routing to **Kling, Veo, Sora, Gemini** behind user-facing tiers (Spotlight/Limelight/Premiere) with fallback and capability detection.
- Built a multi-character continuity engine tracking emotional state, last action, and rolling plot summaries — cached in **Redis**, persisted in **PostgreSQL** — for 20+ segment stories with stale-segment propagation.
- Implemented a serverless backend on **AWS Lambda** (Python + **FastAPI** + **Mangum**) with modular service classes, **SQS** job queue + DLQ, and versioned REST + WebSocket progress streaming.
- Engineered LLM-driven story parsing and character generation that converts free-form text into structured characters, relationships, scenes, and dialogue in one transaction.
- Integrated **Firebase Auth** (Apple/Google/Email, JWT custom claims), **FCM** push, and **Stripe + IAP** credit-based monetization with per-tier subscriptions.
- Provisioned infrastructure with **Terraform** (VPC, RDS, Lambda, SQS, ElastiCache, API Gateway) and **GitHub Actions** CI/CD (Flutter build, Python tests/lint, terraform plan/apply).
- Implemented content moderation (regex + ML input filtering, output scan, human review queue) and GDPR/CCPA data handling (eu-central-1 residency, 30-day hard-delete).
- **Stack:** Flutter (go_router, flutter_bloc), FastAPI, SQLModel, PostgreSQL/asyncpg, Redis, AWS Lambda/SQS/API Gateway/RDS/ElastiCache, Cloudflare R2 + MinIO, Firebase, Terraform, Stripe.

RIDEFLOW — REAL-TIME RIDE-HAILING DISPATCH PLATFORM

- Built a horizontally-scalable real-time dispatch system: stateless **Node.js + Socket.io** servers behind a load balancer with **Redis Pub/Sub** broadcasting driver locations across instances.
- Engineered geospatial nearest-driver matching with **PostgreSQL + PostGIS** `ST_DWithin` (GIST indexes) backed by a **Redis GeoHash** prefix cache for sub-millisecond proximity lookups.
- Designed a 10-state trip lifecycle state machine (REQUESTED → ACCEPTED → ARRIVED → IN_PROGRESS → COMPLETED → PAYMENT_PENDING → PAID) with server-side enforcement, cancellation-fee logic, and payment retry.
- Built a zero-data-loss WebSocket reconnection protocol: monotonic message sequences, exponential backoff with jitter, server-side missed-message replay, and full state snapshot on reconnect.

Ahmad Khidir

- Implemented adaptive GPS polling (2s on-trip → 30s stationary) for battery efficiency plus map marker clustering and viewport culling for 60 FPS at 100+ markers.
- Built a fare engine with surge pricing (zone-based supply/demand multipliers 1.0×–4.0×), cancellation fees, and per-city configurable economics.
- Developed **Flutter** rider + driver apps (Riverpod, go_router, socket_io_client) sharing a `rideflow_shared` core package.
- Planned a multi-city simulation engine with synthetic drivers, realistic route-following, and cancellations across 5 cities (~200 drivers/city).
- **Stack:** Flutter, Node.js/Express/Socket.io, PostgreSQL + PostGIS, Redis 7, React admin dashboard, Mapbox/OSRM, Docker Compose.

GOGATEWAY — HIGH-PERFORMANCE API GATEWAY IN GO

- Built a production-grade reverse proxy from scratch in **Go** using only the standard library (`net/http` + `httputil.ReverseProxy`) with just 3 external deps.
- Implemented **JWT** auth (HS256/RS256, issuer whitelisting, claim forwarding via `X-User-ID`/`X-User-Claims`) plus API-key auth with SHA-256 hashed key stores, tiers, and revocation.
- Engineered a **Redis**-backed rate limiter (atomic Lua script) with automatic in-memory fallback and standard `X-RateLimit-*`/`Retry-After` headers.
- Built a three-state circuit breaker (Closed → Open → Half-Open) with per-route failure thresholds and probe-based recovery.
- Implemented pluggable load balancing (round-robin, least-connections) across multi-upstream routes with wildcard path matching.
- Created a custom **Prometheus** metrics exporter with zero client-library dependencies (request counters, latency histograms, active-connection gauges) plus JSON structured logging (`log/slog`) and `X-Request-Id` tracing.
- Ship-ready: multi-stage **Dockerfile** (~5.4MB Alpine image), **Kubernetes** manifests (2-replica Deployment, probes, ClusterIP, ConfigMap), and docker-compose local dev.
- Wrote comprehensive table-driven test suites with `go test -race` concurrency verification.
- **Stack:** Go 1.22, net/http, Redis, Prometheus, Docker, Kubernetes.

UINLP — MULTI-MODAL ANNOTATION PLATFORM

- Built a full-stack platform for collecting, distributing, and managing multi-modal annotation tasks (text/image/audio/video) at scale on AWS.
- Built a **FastAPI** backend on **AWS Lambda via Mangum** with **Cognito** OAuth (Authorization Code + PKCE) token validation against the JWKS endpoint, fronted by API Gateway.
- Implemented presigned-**S3**-URL dataset upload/download flows and annotator publish workflows with acknowledgment and admin verification.
- Wrote a batch-processor Lambda that extracts uploaded dataset ZIPs, splits files into per-modality chunks, uploads to S3, and flags datasets complete.
- Built an offline-first **Flutter** annotator app (BLoC, amplify_auth_cognito) with dynamically rendered annotation fields and ZIP-based result upload via presigned URLs.
- Delivered a **Next.js 15** admin dashboard (TanStack Query, oidc-client-ts, Zod, repository pattern) for dataset/asset/publish management with soft-delete and verification.
- Provisioned **Terraform** IaC across DynamoDB (3 tables), S3 (3 buckets), Cognito, API Gateway, ECR, and Route 53; shared Pydantic models + repositories in a reusable internal Python library.
- **Stack:** Python 3.12, FastAPI, Mangum, Flutter, Dart, Next.js 15, React 19, DynamoDB, S3, Cognito, API Gateway, Lambda, ECR, Route 53, Terraform, BLoC.

PATIENT & HOSPITAL MANAGEMENT SYSTEM (PHMS)

- Designed a multi-tenant hospital management SaaS digitalizing the full clinical and revenue cycle: registration → billing → service fulfilment → discharge.
- Built a split-identity multi-tenancy model (global users + per-tenant staff memberships) enabling cross-hospital staff to switch context without re-authentication while preserving tenant isolation.
- Engineered dual-layer tenant data isolation: mandatory `tenant_id` query scoping plus database row-level security, with cross-tenant access attempts triggering critical audit alerts.
- Specified an audit-first architecture with append-only audit logs capturing user, tenant, role, module, action, and old/new values synchronously.
- Designed a financial-grade billing engine: auto-bill generation from service orders, tariff catalogues, HMO/NHIS co-pay rules, multi-mode payments, and reversible-only billing with mandatory second-level authorization.
- Engineered race-condition-safe critical paths with `SELECT FOR UPDATE` locking on bill creation, payment posting, and pharmacy stock deduction (no duplicate bills, double-posting, or negative stock).

Ahmad Khidir

- Modeled pharmacy dispense with FEFO (first-expired-first-out) batch ordering, atomic stock deduction, partial-dispense with auto re-prescription, and system-event-only stock mutations.
- Specified clinical workflow: SOAP consultation records, order entry, payment-gated service queues, result review with mandatory-reason amendment, and visit lifecycle state machine.
- Defined production architecture: **PostgreSQL** + read replica (OLTP/OLAP separation), **Redis**, S3-compatible object storage, magic-link auth with account lockout, NDPR/NDPA 2023 compliance, and 30-day backups with PITR.
- Produced a complete spec package: 30 JSON Schema 2020-12 entities, 49 prioritized features (FEAT-001–049) with acceptance criteria, and dev handoff guides.
- **Stack (documented):** React SPA, PostgreSQL (+RLS), Redis, S3, AWS af-south-1, Paystack, Mailgun/SendGrid, WebSocket/SSE.

MED-CLARITY — MEDICAL PAPERWORK AI EXPLAINER

- Built a pan-African web app that turns photographed/pasted medical paperwork into plain-language, drug-data-grounded, multilingual, audio-enabled explanations (explicitly non-diagnostic).
- Architected a service-oriented capture-to-explanation pipeline as independent **FastAPI** services (document-processing, drug-grounding, guardrail, explanation, translation-audio, account, reporting) behind a single **GCP API Gateway**.
- Implemented services with **SQLAlchemy + Pydantic v2** and self-hosted **Tesseract** OCR, LLM-based structured extraction with per-field confidence, a user-correction flow, and mandatory low-confidence disclosure.
- Designed a dual-tier LLM system: **Google Gemini** for routine generation and **Anthropic Claude** for guardrail-critical post-checks enforcing the non-diagnostic scope boundary.
- Built ground-truth drug grounding against reference data (RxNorm/openFDA/WHO Essential Medicines List) with no-match/partial disclosure instead of hallucination.
- Set up a multilingual/audio pipeline: Google Cloud Translation + Cloud TTS for high-resource languages and self-hosted **OmniVoice** TTS (600+ languages) for low-resource African languages.
- Provisioned **Terraform** IaC (Cloud Run/GKE, API Gateway, IAM, per-env state) + **GitHub Actions** CI/CD, Docker Compose local dev, **Neon** serverless PostgreSQL (5 schemas incl. `jsonb` events).
- Added **Firebase Auth** phone-OTP with a guest-first core flow and role-based pharmacy staff access; monetization = free core + **Paystack** B2B2C tier + API licensing.
- **Stack:** React 19 / Next.js 16, TypeScript, FastAPI, PostgreSQL (Neon), GCP, Gemini, Claude, Tesseract, OmniVoice, Firebase Auth, Terraform, Vitest, pytest.

PANTRY-PILOT — AI VIRTUAL KITCHEN APP

- Built a mobile-first virtual home kitchen app unifying pantry inventory, meal planning, recipes, and guided cooking to cut food waste.
- Developed a cross-platform **Flutter** app (**BLoC**, dio, go_router) with **AWS Amplify/Cognito** auth, local notifications, audio alarm, and vibration; cross-target (Android/iOS/web/desktop).
- Architected a **FastAPI** backend (SQLModel, Alembic, mangum) deployed as a Docker-based **AWS Lambda** behind API Gateway.
- Integrated **OpenAI** to translate recipe ingredients into structured JSON for AI pantry-coverage analysis and shopping-gap generation, with rule-based validation rejecting malformed AI payloads.
- Provisioned the full AWS stack in **Terraform**: **Aurora PostgreSQL** (Serverless v2, encrypted, auto-pause), Cognito user pool (PKCE OAuth), Secrets Manager, ECR lifecycle, and a Lambda `db_migrator` running Alembic migrations at deploy time.
- Designed a hybrid recipe ownership model (global catalog + account-scoped favorites + user recipes gated to a paid tier).
- Planned expiry-aware inventory (use-soon surfacing, low-stock flags, leftovers) and guided cooking with per-step timers and substitution hints.
- **Stack:** Flutter/Dart, FastAPI, SQLModel, Aurora PostgreSQL, OpenAI, AWS (Lambda, API Gateway, Cognito, Aurora, Secrets Manager, ECR), Terraform, Alembic, Docker.

KINGY-AUTOS — LUXURY CAR DIGITAL SHOWROOM

- Built an immersive luxury-car dealership “digital showroom” web app (50–100+ vehicle catalog) for high-net-worth buyers with cinematic 3D/motion presentation.
- Developed in **Next.js 16 App Router + TypeScript (strict)** with a cinematic homepage, filterable inventory catalog (grid/list views), full-text search with autocomplete, and SEO-rich car detail pages.
- Implemented **Three.js** hero scenes via react-three-fiber/drei (MeshReflectorMaterial, Sparkles, 2D-in-3D photo composition) with async lazy loading, reduced-motion fallback, and gyroscope/mouse parallax.
- Integrated headless **Sanity CMS** (GROQ queries, typed schema, ISR webhook revalidation) with server API routes for inquiry, test-drive booking, and search.
- Engineered the conversion flow with zod + react-hook-form validation, WhatsApp click-to-chat, favorites, and a 3-car comparison tool persisted in local storage.

Ahmad Khidir

- Designed a JAMstack architecture: SSG/ISR static pages, **Cloudinary** image CDN (WebP/AVIF, blur-up, srcset), privacy-first analytics, and server-side-only email/WhatsApp notifications.
- Authored the full engineering doc suite (architecture, JSON Schema 2020-12 data model, features, screens, monetization, risk log) plus SEO foundation (JSON-LD Vehicle schema, sitemap, OG tags) and WCAG 2.1 AA dark-mode-first accessibility.
- **Stack:** Next.js 16, TypeScript, Tailwind v4, Sanity CMS, Three.js/R3F/drei, Cloudinary, zod, react-hook-form, pnpm, Vercel.

SHOPHIVE — SERVERLESS MULTI-TENANT E-COMMERCE PLATFORM

- Architected a fully serverless multi-tenant e-commerce-as-a-service platform where sellers launch branded storefronts on subdomains with unified checkout, chat, and search.
- Built **AWS Lambda (TypeScript + Hono)** services, **API Gateway** (REST + WebSocket), **DynamoDB** (16 entities, on-demand), **S3**, **CloudFront**, and **Cognito** wired with SNS/SQS/EventBridge async messaging.
- Built a **Go Lambda@Edge** subdomain router (~50ms cold start) mapping `{store}.shophive.net` → S3 directory with no per-store DNS, supporting custom domains via ACM + CNAME polling.
- Implemented an order escrow state machine on **Paystack** (card/bank/USSD, NUBAN payouts, subaccounts, HMAC-SHA512 webhook + SQS DLQ) with Flutterwave fallback, 14-day auto-release, and dispute/refund flows (money in integer kobo).
- Delivered hybrid semantic search via **OpenAI text-embedding-3-small / Bedrock Titan** into an S3 vector index, synced automatically via **DynamoDB Streams**, with similar-products recommendations.
- Built an event-driven storefront pipeline: SNS-triggered Build Lambda runs **Next.js** static export with per-product `generateStaticParams`, uploads to S3, and invalidates CloudFront — live in <5 min with zero downtime on failure.
- Implemented unified cart, per-store checkout, real-time **WebSocket** buyer↔seller chat (WhatsApp/SES offline notifications), discount codes, inventory, variants, digital products, and reviews.
- Hardened with WAF, PCI-scope isolation (checkout Lambda VPC egress-only to Paystack), KMS encryption, Secrets Manager, PITR backups, and rate limits (1,000 req/s platform, 100/s per IP).
- **Stack:** TypeScript, Hono, Node.js 20, Go 1.22, Next.js 15, DynamoDB, S3, CloudFront, Lambda@Edge, Cognito, Paystack, Flutterwave, OpenAI, Bedrock, WhatsApp, SES, Terraform, GitHub Actions, Zod.

VANGUARD-BANK — DIGITAL BANKING PLATFORM

- Designed a full-stack digital banking platform for retail and SMB users: **Flutter** mobile app + 6 **Go** microservices with event-driven async architecture.
- Built microservices with bounded contexts (auth, account, transfer, card, notification, analytics) behind a **Go + Chi** API gateway; REST for clients, **gRPC/Protobuf** for inter-service calls, **NATS JetStream** pub/sub event bus (7-day retention, DLQ, 5 max deliveries).
- Implemented banking-grade auth: **JWT RS256** (15-min access / rotated 7-day refresh), **TOTP MFA** (RFC 6238), bcrypt (cost 12), device binding, session expiry, and rate-limited endpoints.
- Built transfer orchestration with **Redis**-backed idempotency keys (24h TTL), dual-approval workflows for business accounts, standing orders, batch payments, and atomic balance operations.
- Created a virtual card service (Luhn-valid, ISO/IEC 7812 PANs, encrypted CVV with biometric-verified reveal, per-card spending limits, lifecycle states) with clipboard auto-clear after 60s.
- Implemented push notifications (**FCM**/APNs) with per-category preferences and a fraud-signal analytics service consuming the event stream; all money stored as integer minor-currency units.
- Enforced Clean Architecture on Flutter (Presentation→Domain→Data, Bloc/Cubit, GoRouter, token-based design system, certificate pinning) and the full testing pyramid (unit, widget, golden, integration, smoke, performance).
- Delivered observability via `slog` JSON logging, **Prometheus** metrics, and **OpenTelemetry** tracing, with Docker Compose reproducibility and **GitHub Actions** CI (lint → test → build → containerize).
- **Stack:** Flutter, Dart, Go 1.22, Chi, gRPC, Protobuf, NATS JetStream, PostgreSQL 16, Redis 7, golang-migrate, OpenAPI 3.0, Prometheus, Grafana, OpenTelemetry, Docker.

FAMILY KITCHEN — SMART KITCHEN MANAGEMENT

- Built a mobile-first kitchen management system combining household inventory, meal planning, and chef workflows under a playful "Kingdom" metaphor for families.
- Built a **FastAPI** backend (Authlib OAuth2, JWT, bcrypt) on **GCP Cloud Run** with **Firestore** covering auth, kingdom/member CRUD, inventory, allergy vault, and recipe endpoints.
- Implemented hierarchical **RBAC** (Visitor → Worker → Prince/Princess → King/Queen, cumulative permissions) enforced on all routes with role-based endpoint protection.
- Designed inventory with expiration intelligence: `best_before` tracking, `max_use_count`/`total_use_count`, and configurable expiring-soon alerts.
- Built allergy-vault features: per-user allergen registry, known-allergen database, and recipe-allergen conflict detection that blocks recipe creation against kingdom members.

Ahmad Khidir

- Engineered OAuth2 + JWT security: 15-min access / 7-day refresh tokens with JTI-based blacklisting, refresh rotation, audit logging, and encrypted PII at rest (AES-256-GCM via Secret Manager).
- Provisioned GCP infrastructure with **Terraform** (Cloud Run, Firestore indexes + security rules, Secret Manager, remote GCS state) and containerized deployment via Docker.
- Flutter mobile app with **SQLite (sqflite)** offline cache + sync manager, **FCM** push, **google_mlkit** barcode scanning, and workmanager background sync.
- **Stack:** FastAPI, Authlib, python-jose, Firestore, Flutter/Dart, GCP Cloud Run, Terraform, Docker, pytest + mockito.

INTERLACE — AI UI DESIGNER & FIGMA PLUGIN

- Built a **Figma plugin** in TypeScript that renders CSS styles as native Figma nodes, with resolvers for px/rem/vw units, colors, gradients, flexbox layout, box-shadow/filter blur, and typography.
- Implemented Figma node creators for `<img>`, `<video>`, `<button>`, and `<input>` (text/checkbox/radio/range/color) with plugin-data persistence and static-image fallbacks.
- Set up an esbuild + React-JSX widget build pipeline (watch mode, ESLint, tsc type-check).
- Scaffolded a **Next.js 16** + **Tailwind CSS v4** web app with **Zustand** state management.
- **Stack:** TypeScript, Figma Plugin/Widget API, esbuild, csstype, React JSX, Next.js 16, Tailwind CSS v4, Zustand.

SKELA — THREE.JS OBJECT/SCENE FRAMEWORK

- Designed a reusable **Three.js** framework for composing 3D objects and scenes via a clean class hierarchy with zero build tooling.
- Designed `BaseObject` with a lifecycle interface (`build/update/render/dispose`) and a `perform()` action dispatcher making position, rotation, scale, and visibility first-class actions.
- Created a `base.jinja` scene template with Jinja block inheritance handling renderer, camera, lighting, and orbit controls.
- Built a reference `Human` subclass demonstrating a skeletal hierarchy with a multi-channel animation/action system.
- Developed a companion **Next.js 16 + React Three Fiber** website with an interactive humanoid supporting walk/run/kick/jump/wave/punch/dance/nod/bow/spin modes.
- **Stack:** Three.js, JavaScript, Jinja2, Flask, Python, Next.js 16, React 19, @react-three/fiber, @react-three/drei, Tailwind 4, TypeScript.

TEXT-TO-CAD — PROGRAMMATIC 3D MODEL GENERATION

- Set up a text-to-CAD harness for generating source-controlled 3D models using coding agents (Python CAD runtime on **build123d/OpenCascade**).
- Produced STEP, STL, DXF, GLB, and topology outputs; URDF robot descriptions via a bundled URDF skill.
- Vendored agent skills (CAD + URDF) exposing CLI tools (`gen_step_part`, `gen_step_assembly`, `gen_urdf`, `cadref`, `snapshot`) for reproducible regeneration workflows.
- Implemented `@cad[...]` prompt-reference handles for precise geometry-aware follow-up edits with stale-ref handling.
- Authored custom parametric models (chemistry burette assembly, drinking bottle) with checked-in generated artifacts.
- Built a local **React 18 + Three.js + Vite** CAD Explorer viewer (Tailwind 4, radix-ui) with `node:test` unit tests for geometry, URDF parsing/kinematics, DXF parsing, and scene scale logic.
- **Stack:** Python, build123d, OpenCascade, ezdxf, trimesh, numpy, vtk, React 18, Three.js, Vite 7, TypeScript, URDF.

BREWCRAFT COFFEE — SCROLL-STORYTELLING WEBSITE

- Built an immersive scroll-storytelling marketing website giving a local coffee brand its first online presence (inquiry-based ordering: contact form, WhatsApp, phone).
- Developed with **Astro** static generation, strict TypeScript, and **Tailwind v4** CSS-first `@theme` tokens (single theming system, light/dark pairs).
- Engineered a single scroll-progress motion driver (**Lenis** + **GSAP ScrollTrigger**) powering 0.3/0.6/1.0 parallax layers and ~150vh pinned story sections, with reduced-motion as a first-class path.
- Integrated **Sanity CMS** (embedded Studio, GROQ queries, idempotent seed script) so a non-technical owner edits products/story/photos without a developer.
- Implemented **Web3Forms** contact submission with confirmation states, WhatsApp deep links, click-to-call, and availability badges.
- Enforced WCAG AA accessibility (44×44px targets, focus rings, aria-describedby) and a <1.5MB payload / 60fps scroll performance budget with WebP srcset imagery.
- Set up **Vitest** unit tests, **Playwright** E2E + axe-core a11y tests, and husky/commitlint/lint-staged hooks enforcing Conventional Commits.
- **Stack:** Astro 7, TypeScript, Tailwind 4.3, GSAP, Lenis, Sanity 6, Web3Forms, sharp, Vitest, Playwright, Vercel.

NARRA-AI — AI NARRATIVE PRODUCTION PLATFORM

- Designed an AI narrative-production platform that turns story premises into segmented multimedia stories (narrative text + voice narration + cinematic stills) while preserving character/plot/tone continuity.
- Built the core differentiator: a persistence/continuity memory layer plus a media orchestration layer aligning word-level TTS timestamps with beat markers and scene cues (DTW-style temporal validation, modality-specific regeneration to cut cost).
- Built a working prototype narrative engine in **Python** that drives an LLM with strict JSON Schema-aligned prompts to emit story plans (story, characters, segments with beats/sceneCues/targetDurationMs, aggregate storyState).
- Prototyped video generation with **Google Veo 3.1 Fast** via the genai SDK, composing image-reference assets with timestamped scene-cue prompts into 8-second clips.
- Authored a 24-document engineering/product suite: architecture (Mermaid sequence diagrams), data model, features, screens, user flows, content moderation, legal/IP, localization, testing, analytics.
- Defined credit-based monetization with per-segment unit economics and cost levers (caching, resolution tiers, provider fallback, audio compression).
- **Stack:** Python, OpenAI-compatible client (LLM planning), Google genai (Veo 3.1), JSON Schema 2020-12, Mermaid, MP4.

AGRILINK — B2B AGRITECH SUPPLY-CHAIN PLATFORM

- Designed a B2B agritech supply-chain aggregator connecting Nigerian farming cooperatives to urban bulk buyers, eliminating predatory middlemen.
- Designed a multi-modal platform: farmers via **USSD/SMS (AfricasTalking)**, buyers via a **React** web dashboard, drivers via an **Android** app with offline route manifests.
- Architected domain-partitioned services (Matching Engine, **VRP Route Optimization**, Escrow, Pricing, Notification) over **PostgreSQL**, **Redis**, **S3**, and a durable job queue (at-least-once, retries).
- Spec'd payments via **Paystack/Flutterwave** (escrow) with payouts to mobile money, buyer verification via CAC API, and NIN/BVN cooperative identity; AES-256 at rest, NDPR compliance.
- Built-in localized flows (Hausa, Yoruba, Igbo, Pidgin), live delivery tracking via **Server-Sent Events**, QR batch-graded produce tags, dispute resolution, and an ops admin panel with MFA.
- **Stack (documented):** USSD/SMS, React, Android, Google Maps/OSRM, PostgreSQL, Redis, S3, Paystack/Flutterwave, AES-256, SSE.

EDUCATION

B.Sc. Computer Science | University of Ibadan, Nigeria | 2025

SKILLS

- **Languages:** Python, JavaScript, TypeScript, Go, Dart
- **Backend:** Django, FastAPI, Node.js, Express, Hono, Serverpod, Socket.io, gRPC, WebSockets, REST, GraphQL, Celery
- **Frontend:** Flutter, React, React Native, Next.js, Astro, Tailwind, Three.js, React Three Fiber, BLoC, Riverpod, Provider, Redux
- **Cloud & DevOps:** AWS (Lambda, ECS, Fargate, DynamoDB, RDS, Aurora, S3, Cognito, API Gateway, Lambda@Edge, SQS, WAF), GCP (Cloud Run, Firestore, GCS, Cloud Tasks), Docker, Kubernetes, Terraform, CI/CD, Serverless
- **AI/ML:** OpenAI, Google Gemini, Anthropic Claude, Vertex AI, Bedrock, pgvector, Veo, Tesseract OCR, LLM orchestration, Semantic Search, Embeddings
- **Data & Storage:** PostgreSQL, PostGIS, pgvector, MySQL, Redis, Firestore, DynamoDB, SQLite, DuckDB, S3, GCS, Cloudflare R2
- **Payments:** Stripe, Paystack, Flutterwave, escrow, subscription billing, idempotent webhooks, HMAC signature verification
- **Auth & Security:** OAuth2, JWT, Firebase Auth, Cognito, RBAC, TOTP MFA, device binding, rate limiting, encryption at rest, NDPR/GDPR
- **Tools:** Git, GitHub Actions, Prometheus, OpenTelemetry, Grafana, k6, Jest, Vitest, Playwright, pytest, Terraform, Docker, Unix