Tran Huy Hoang

FULL-STACK, AI & MOBILE ENGINEER · SOFTWARE ARCHITECT

Ho Chi Minh City, Vietnam (UTC+7)

(+84) 93-XXX-XXXX | tranhuyhoangstu@gmail.com | github.com/hoang-IT-eng | portfolio

"Engineering systems where domain logic, offline resilience, and AI hold up under real-world load."

Summary

Full-Stack, Computer Vision & Android engineer shipping offline-first mobile apps, NestJS/.NET commerce APIs, and biometric attendance pipelines. Comfortable owning the path from schema and auth contracts through React/Vite storefronts — with Clean Architecture boundaries, JWT sessions, and optional Redis caching that degrades safely when Redis is absent.

Work Experience & Shipped Systems

IronLog (GYMVERSE) — Offline-First Android Fitness Log

Saigon, Vietnam

Android Engineer · Kotlin, Jetpack Compose, Room, Foreground Service *2025*

`CASE` **Selected Work — IronLog** `REPO` **github.com/hoang-IT-eng/GYMVERSE**

- Used Room SQLite as the write-ahead source of truth so workout sets and routines survive airplane mode and process death without a network dependency.
- Ran the rest timer as an Android foreground service with a live notification so the countdown keeps counting when the phone locks between sets.
- Drove Compose screens from Kotlin StateFlow so UI state mirrors Room rather than inventing in-memory progress that disappears on rotation or kill.

Attend — Anti-Spoof Face Attendance Pipeline

Saigon, Vietnam

AI & Backend Engineer · Python, FastAPI, DeepFace, MediaPipe, WebSockets *2025*

`CASE` **Selected Work — Attend** `REPO` **github.com/hoang-IT-eng/AI-Face-Recognition-Attendance**

- Gated recognition behind a MediaPipe blink liveness checker (two blinks required) so a printed photo cannot pass attendance before Facenet512 runs.
- Streamed the camera loop over FastAPI WebSockets (`/ws/liveness`) so the kiosk UI gets live liveness status and only then matches Facenet512 embeddings.
- Ran three-angle enrollment in `RegisterFace.jsx` (stronger detectors for those captures) and a lighter path for continuous monitoring on the React dashboard.

Atelier (THEBASIC) — Fashion Storefront + NestJS API

Saigon, Vietnam

Full-Stack Engineer · React 19, Vite, NestJS, TypeORM, Postgres, Redis, JWT *2026*

`LIVE` **Azure Static Web Apps storefront** `CASE` **Selected Work — Atelier** `REPO` **fe-fashion-shop**
`REPO` **be-fashion-shop**

- Added optional Redis caching for catalog reads that disables itself when `REDIS_URL` is missing or unreachable — cutting repeat product hits without hard-failing the API on free tiers.
- Took pessimistic write locks inside order transactions so concurrent checkouts cannot oversell the same stock row while VNPay return handlers update payment state.
- Issued a Passport JWT with `expiresIn: '7d'` (no refresh token). bcrypt hashes passwords only; on 401 the React `apiClient` clears the session and logs the shopper out.

Layers (TaskManagement_System) — Clean Architecture Task API

Saigon, Vietnam

Full-Stack Engineer · ASP.NET Core, EF Core, SQL Server, FluentValidation, React *2025*

`CASE` **Selected Work — Layers** `REPO` **TaskManagement_System**

- Split Domain / Application / Infrastructure / API so business entities never reference ASP.NET — new features land as use cases instead of controller-side SQL.
- Persisted ActivityLog rows with indexed entity/user/time columns so comments and status changes leave an auditable history instead of silent mutations.
- Validated inbound DTOs with FluentValidation and authenticated callers with JWT Bearer so invalid payloads and anonymous writes fail at the edge with consistent contracts.

Plates — License Plate Detect + OCR Pipeline

Saigon, Vietnam

AI Engineer · Python, YOLOv8, EasyOCR, OpenCV

2024

CASE **Selected Work — Plates** REPO **License-Plate-Recognition-System**

- Ran plate detection with stock YOLOv8 nano / OBB weights (no fine-tuned checkpoint committed), then cropped and fed EasyOCR so the product optimizes for readable text — not just a high detection score.
- Kept a training script plus `evaluate_ocr.py` in-repo so OCR accuracy is measured character-level against labeled crops instead of trusting a demo GIF.

Supply — Stationery Shop Gateway · Identity · Commerce

Saigon, Vietnam

Full-Stack Engineer · Next.js, NestJS, API Gateway, SQL Server, JWT

2025

CASE **Selected Work — Supply** REPO **web-quan-ly-van-phong-pham *(Private)***

- Put an API gateway in front of identity (:3001) and commerce (:3002) so the browser only calls relative `/api` and `/uploads` — a microservices boundary without leaking internal ports.
- Owned JWT access/refresh on identity with a client-side refresh queue in `frontend/js/api.js`: on 401 the storefront refreshes once, queues concurrent callers, and replays so checkout does not bounce the shopper out.
- Split `identity_db` and `stationery_db` on SQL Server 2022 so auth schema changes cannot corrupt catalog/order tables.

Counter — Coffee POS + React Twin

Saigon, Vietnam

Full-Stack Developer · ASP.NET Core, Identity, SQL Server, SignalR, React, Vite

2023

CASE **Selected Work — Counter** REPO **CoffeeShopSolution** REPO **CoffeeShopFrontend**

- Modeled tables, menus, orders, and bills in SQL Server first so the ASP.NET Core + Identity API and React operate UI are projections of a normalized schema under rush-hour edits.
- Pushed kitchen/order updates through a SignalR `KitchenHub` so the React twin stays in sync without polling the ASP.NET Core API.

Folio — Interactive Portfolio + Demo Experiments

Saigon, Vietnam

Design Engineer · React, Vite, Framer Motion · Three.js/GSAP on personal folio

2026

LIVE **hoang-dev-portfolio.netlify.app** CASE **Selected Work — Folio** REPO **demo-portfolio**

REPO **hoang-portfolio *(Private)***

- Shipped `demo-portfolio` as React + Vite + Framer Motion experiments for layout and interaction — no Three.js/GSAP in that repo.
- Carried Three.js studio craft and GSAP scroll into the private `hoang-portfolio` twin, structuring Selected Work as narrative journeys (challenge → approach → backend notes).

Core Technical Stack

Backend & Architecture:	ASP.NET Core, C#, NestJS, TypeScript, Python, FastAPI, Clean Architecture, REST APIs, JWT access/refresh, FluentValidation, Docker.
AI & Computer Vision:	DeepFace (Facenet512), MediaPipe blink liveness, OpenCV, YOLOv8, EasyOCR, WebSocket camera loops.
Mobile & Persistence:	Kotlin, Jetpack Compose, Room SQLite, Foreground Services, Coroutines & StateFlow, PostgreSQL, SQL Server, Redis (optional).
Frontend & Creative:	React, Next.js, Vite, Zustand, Tailwind CSS, GSAP, Framer Motion, Three.js.

Engineering Stance & Principles

• **Separate domain from delivery:** Business rules stay in domain/application layers; controllers only translate HTTP.

• **Liveness before recognition:** Face matching never runs until the blink challenge passes.

• **Offline-first when it matters:** Critical gym interactions commit to Room before any sync fantasy.

• **Cache without hard dependency:** Redis speeds catalog reads but the Nest API keeps serving when Redis is down.

• **Commerce is integration:** Auth, cart, inventory, and VNPay return handling share one transactional story.

Education

Saigon Technology University (STU)

Ho Chi Minh City, Vietnam

Bachelor of Science in Information Technology / Software Engineering *2020 – 2024*

- Core coursework: Data Structures & Algorithms, Object-Oriented Analysis & Design, Database Management Systems, Distributed Systems, Software Engineering.

Languages & Certifications

- **English:** TOEIC 775
- **Vietnamese:** Native