NAJIH Driss

Senior Software Engineer – AI-Driven Apps

France · najih.driss73@gmail.com · +33 7 49 52 96 00
LinkedIn: https://www.linkedin.com/in/najih-driss/
GitHub: https://github.com/kouji-dev
Portfolio: https://www.kouji.dev
Book a meeting: https://calendar.app.google/KTPtgPtkgrbJUEjm7

PROFESSIONAL SUMMARY

Senior Software Engineer with **7+ years of experience** designing and building large-scale web platforms and backend frameworks. Extensive experience across JavaScript/TypeScript, Java, Angular, React, and cloud-native architectures.

Specialized in **complex frontend frameworks, backend systems, and platform-level development**, with experience working on **shared frameworks, low-code-like systems, and enterprise tooling** used by multiple teams and products.

Experienced in **AI-driven application design**, particularly for **backend systems prepared for AI integration**, with hands-on usage of **Python and LLM APIs in personal and proof-of-concept contexts** to design AI-enabled workflows, automation, and agent-based capabilities.

Strong focus on clean architecture, scalability, maintainability, and developer experience.

AI & PYTHON CAPABILITIES

- Use of **Python as a general-purpose language** for AI-driven applications, automation, and backend services (personal and PoC contexts)
- Hands-on experimentation with **LLM APIs**
- Design and testing of **AI agents**, prompt engineering strategies, and agent workflows
- Strong interest in integrating AI capabilities into existing enterprise systems rather than isolated ML projects

TECHNICAL SKILLS

Languages: JavaScript, TypeScript, Java 8/11/17/21, Python (AI/automation – PoC), Rust, Go

Front-End: Angular (15 → 22), RxJS, NGRX, NGXS, Angular Material, React, Redux, TailwindCSS, Ant Design, MUI, HTML5, CSS3/SCSS, Lexical, AG-Grid, ECharts

Back-End: Spring Boot, Spring MVC, Spring Data, Spring Security, Spring Batch, Hibernate, JPA, Node.js, Express.js, NestJS, Prisma, Quarkus

APIs: REST, GraphQL

Databases: PostgreSQL, Oracle DB, SQL Server, MongoDB, MySQL, Redis

Messaging: RabbitMQ, Kafka

Security: OAuth2, JWT, Keycloak, Azure AD (SSO)

Cloud: AWS (Lambda, Step Functions, S3, DynamoDB, EC2, Elastic Beanstalk, CloudFront), Azure (AD, App Services, CDN, Front Door, Blob Storage, PostgreSQL), OVHcloud

DevOps: Docker, Kubernetes, GitLab CI/CD, Jenkins

Testing: JUnit 5, AssertJ, Mockito, WireMock, Jest, Karma, Testing Library, Cypress, Playwright, JaCoCo, SonarQube

Tools & Practices: Agile Scrum, Git, GitHub, GitLab, Jira, Maven, NPM, Yarn, IntelliJ, Figma, TDD, DDD, BDD, Clean Architecture, Hexagonal Architecture

PROFESSIONAL EXPERIENCE

Amundi ITS — Senior Full Stack Developer (Angular / Java)

September 2023 – Present (3 years)

Project context: Amundi ITS develops shared platforms and frameworks used by multiple internal and client-facing applications. A major product is the **Maestro framework**, an Angular-based solution enabling backend teams to rapidly integrate consistent, enterprise-grade user interfaces.

Framework Team

- Maintained and evolved the **Maestro Angular framework** used by multiple internal and client-facing applications
- Actively participated in **major Angular version migrations (15 → 18 → 20)**, including component upgrades, dependency updates, and adoption of new framework capabilities (Signals, Standalone Components, etc.)
- Designed and developed **major reusable Angular components** integrated into the framework (forms, data grids, navigation, layouts, etc.)
- Co-designed the **framework architecture** for compatibility with **micro-frontend environments**, enabling smoother integration into modular and distributed applications
- Managed versioning and delivery of framework releases, ensuring stability for consuming teams
- Ensured component quality, accessibility, and adherence to Angular best practices
- Provided **level 3 support** to application teams integrating the framework
- Partnered with product owners and UX designers to translate business needs into scalable UI components

Lowcode Team

- Delivered a **drag & drop, no-code interface** enabling teams to compose screens visually without writing code
- Created the **low-code library** powering the Web Designer — a **descriptor-pattern engine** where screens, components and behaviors are defined declaratively via structured **JSON**
- Redesigned a rigid declarative system into a **more scalable and maintainable architecture**
- Maintained and evolved the low-code platform (bug fixes, refactoring, performance improvements, new features)
- Led or supported the **migration of three existing client applications** onto the low-code platform, acting as the platform-side referent for each team
- **Siebel migration (ESR team)** — migrated an **Oracle Siebel legacy application** onto the low-code platform, rebuilding its screens as declarative descriptors
- **Alto team** — migrated the application from the **legacy descriptor format to the new descriptor architecture**, defining the migration path and supporting the team through breaking changes
- **Dashboard** — delivered the application as the **Web Designer candidate**, validating the designer end to end from visual composition to generated JSON descriptors
- Also re-platformed a **legacy Java desktop client to the web** and moved critical business components (Employee Savings Plan) off legacy tooling
- Contributed to the **architecture of client applications** from other teams, providing technical expertise and best practices

- Built the **Web Designer from scratch** with an **AI agent** for **natural-language generation of web applications** — describe a screen, it scaffolds the JSON-defined views, components and bindings
- Defined the product **UX in close collaboration with the Product Owner**, shaping user flows and interaction patterns end to end

AI Tooling Team — Vibe Toolbox (current)

- Building the **Vibe Toolbox**, an internal **Electron + Angular** desktop app **provisioning AI-ready dev environments** (agent skills, MCP servers), cutting time-to-productivity from days to minutes
- Implemented **skills management** — installation, versioning and sharing of reusable AI skills (prompts, workflows, agent definitions) so teams adopt validated patterns instead of rebuilding them per project
- Built **marketplace mechanics** for AI skills and agents, supporting discovery, packaging and lifecycle management with role-based governance suited to enterprise use
- Designed the toolbox as a **single entry point** for developers to bootstrap model access, credentials, MCP configuration and skill bundles, bringing consistency to how AI tooling is adopted across teams

Cross-team contributions

- Designed, built and published multiple **internal Java and JavaScript packages** (libraries, shared utilities, framework modules) consumed by Maestro and downstream application teams
- Set up and maintained **automated CI/CD pipelines** (GitLab CI, Jenkins) covering build, lint, tests, coverage and release of framework and low-code artifacts
- Ensured **strong test coverage** across frontend and backend with **JaCoCo** (Java), **Karma** and **Jest** (Angular/TypeScript), plus **Cypress** and **Playwright** for end-to-end scenarios
- Daily use of **agentic AI tooling** (Claude, Codex) for code generation, refactoring, code reviews, debugging, and accelerating exploratory work on framework and platform features

Technical Challenges:

- Micro-frontend integration and **framework backward compatibility** across major Angular version migrations
- Replacing a rigid declarative legacy system with a **scalable descriptor-based low-code engine** without breaking existing client applications
- Designing framework APIs **generic enough for many teams** yet simple enough for rapid adoption

Technical Environment: *Angular, TypeScript, Node.js, Electron, SQLite, PowerShell, Java 8/11/21, Spring Boot, Quarkus, Keycloak, Oracle DB, MongoDB, Karma, Jest, JaCoCo, Cypress, Playwright, Webpack, Jenkins, GitLab CI, Claude, Codex, Agile Scrum*

Zsoft Consulting — Full Stack Consultant (Java / React)

October 2022 – September 2023 (1 year)

Project context: IT consulting firm. Full-time consultant delivering successive missions for major clients (Société Générale, Colas).

Client: Société Générale — Project: KYC

Project context: A set of backend services dedicated to **Know Your Customer (KYC)** processes, used to evaluate customer trust levels and eligibility, and to support fraud detection workflows.

Responsibilities:

- Led a **full-backend mission to extract a standalone, reusable microservice** from a legacy **KYC** system, consumable across the organization
- Decoupled the service from a **database owned by another provider team**, defining clean service boundaries and ownership
- Designed and implemented **backend services** with Java and Spring Boot
- Developed and exposed REST APIs consumed by other internal systems
- Implemented security mechanisms using Spring Security and OAuth2
- Provided **L2 support**, investigating and resolving production incidents
- Ensured quality with unit and integration tests plus **code coverage (JaCoCo)**, for reliability and compliance

Technical Challenges:

- Extracting a standalone service from a **tightly coupled legacy KYC system** whose database was owned by another provider team

Technical Environment: *Java 17, Spring Boot, Spring Security, OAuth2, PostgreSQL, Quartz, Flyway, JaCoCo, Docker, Kubernetes, RabbitMQ, Jenkins*

Client: Colas — Project: CRM

Project context: CLASS — a **global CRM / Opportunity Manager** built from scratch and deployed across 55 countries to manage commercial activities, opportunities, and customer data for the Colas group, replacing a legacy **Power Apps / Teams** application with a modern React + Java Spring Boot stack.

Responsibilities:

- Built the application **from scratch**, from mockups to production deployment
- Drove the **migration from Power Apps / Teams** to a maintainable React + Spring Boot app, rebuilding the opportunity-tracking workflows (filtering by type, region and other criteria)
- Implemented the UI from designer mockups using **React and the Ant Design system**, with consistent, reusable components
- Implemented secure authentication with **Azure AD (IAM / SSO)**, integrated via Spring Security
- Developed backend services and REST APIs using Java and Spring Boot
- Developed batch processes for **data migration** using Spring Batch
- Deployed the application on **Azure** (App Services, Front Door, CDN, Blob Storage, PostgreSQL) with **CI/CD pipelines**

Technical Challenges:

- Replacing a legacy **Power Apps / Teams** application without disrupting commercial workflows across **55 countries**

Technical Environment: *React, TypeScript, Ant Design, Java 11, Spring Boot, Spring Security, Spring Batch, Azure (AD, App Services, CDN, Front Door, Blob Storage, PostgreSQL), GitLab CI/CD*

Client: gloody.co — Project: HR & Business Management SaaS

Project context: gloody.co — a **multi-tenant HR & business management SaaS** centralizing the day-to-day workflows of client companies: absences and holidays, invoicing, expense reports and meeting-room booking, all served from a single codebase.

Responsibilities:

- Delivered **full-stack features** across the suite — absences & holidays, invoicing, expense reports and meeting booking — with **React / Ant Design** front-ends on **Java / Spring Boot** REST APIs
- Owned the **front-end architecture**, notably a **high-performance calendar** rendering dense scheduling data (absences, holidays, room bookings) with smooth navigation
- Structured the front-end into **reusable Ant Design components** and shared state patterns, keeping the suite's modules consistent
- Designed and exposed **REST APIs with Java and Spring Boot** on a multi-tenant data model, keeping each client organization's data isolated
- Worked on an **AWS-hosted** deployment, contributing to the delivery pipeline

Technical Challenges:

- Keeping the **calendar fluid** while rendering large volumes of multi-source scheduling data on a multi-tenant model

Technical Environment: *React, TypeScript, Ant Design, SCSS, Java, Spring Boot, REST APIs, AWS*

Scor — Full Stack Developer → Project Lead (Java / Angular)

June 2019 – September 2022 (3 years 4 months)

Project context: RiskReveal — a **heavy data-processing application** for **natural-catastrophe risk assessment**, ingesting and processing large files of historical loss events (**ELT — Event Loss Tables**), each tied to a geographic region and a peril type (flood, hurricane, etc.); reinsurers use the data to assess and price the risk of a given portfolio.

Responsibilities:

- Grew from full-stack developer into **technical / project lead**, validating backend (Spring) delivery and driving implementation across the team
- Built the **data-heavy risk UI** on **AG Grid** — virtualised grids, grouping and aggregation over large **PLT loss datasets**
- Owned the **calibration model** — statistics-based calculations applied to statistical loss events stored in **PLT files**
- Developed **statistics dashboards** over PLT loss data — aggregated loss indicators per peril and region, fed by the calibration calculations
- Designed and developed backend services and REST APIs with Spring Boot

Achievements & Impact:

- Optimized and evolved the **batch orchestration system** (Spring Batch) on large-volume risk data, with **WebSockets** for real-time job progress
- Set up **SSO authentication** with Spring Security, integrating **Azure AD** and **Keycloak** as identity providers
- Refactored the frontend onto a **single, unified design system (Ant Design)**, removing legacy **PrimeNG** components
- Made **large risk datasets workable in the browser** — virtualised grid rendering and data visualization on the heavy loss-data screens

Technical Environment: *Java 8, Spring Boot, Spring Batch, Spring Security, Angular 12, Ant Design, AG Grid, WebSockets, Azure AD, Keycloak, SQL Server, Azure DevOps, Jenkins*

Scor — Angular Front-End Developer (End-of-Study Internship)

March 2019 – June 2019 (4 months)

Project context: A pricing management platform for internal pricing teams.

- Developed frontend features using Angular
- Designed and styled UI components
- Integrated REST APIs
- Applied best practices in HTML, CSS, and TypeScript

SIDE PROJECTS

Orrery — Multi-Agent Git Orchestrator IDE (Desktop)

Role: Solo Author — Full Stack Developer · **Type:** Personal Open-Source Project

Repository: https://github.com/kouji-dev/orrery-releases

Project context: Orrery is a futuristic, IntelliJ-inspired **multi-agent git orchestrator IDE**, built as a native desktop application with **Tauri (Rust) and Angular 20**. Each project is a git repository, and a project can **spawn multiple AI coding agents** (Claude Code, Codex, Cursor, Gemini) that each run in their own **git worktree, branch, terminal and conversation** — letting a developer drive several agents in parallel from a single cockpit.

- Architected the application as a **Tauri 2 desktop shell (Rust)** wrapping an **Angular 20** front-end, with a clean **module-per-domain** structure (projects, agents, workspace, orchestrator dashboard, right panel, sidebar, settings, …)
- Built the **orchestrator dashboard** — a hero overview with live stats and multiple visualization metaphors for monitoring agents across projects
- Implemented the **per-agent workspace** with Diff / Terminal / Chat panes: **WebGL-accelerated integrated terminals**, a diff/merge editor on **CodeMirror 6**, and rich-text / markdown editing via **Lexical**
- Designed the **spawn-agent flow** (project, branch, tool, model, effort, prompt) and an **agent-worktree runtime** so each agent is isolated on its own branch and worktree
- Integrated **agent cost tracking** (ccusage) and an **agent hooks / permissions** model to govern what agents are allowed to do
- Built a **design-token-driven design system** with theming, density, colour palettes and motion controls, plus real-time streaming logs and live timers
- Wrote the **Rust backend** pieces (auto-updater, app-icon generation, hooks CLI) and wired the **auto-update** flow via Tauri's updater plugin
- Established a **quality & release pipeline**: **Vitest** unit tests, **Playwright** end-to-end suites, a **performance smoke-test** harness with assertions, semantic version-bump scripts, and **GitHub Actions** workflows (including a Render-deployed landing site)

Technical Environment: *Tauri 2, Rust, Angular 20, TypeScript, CodeMirror 6, Lexical, RxJS, Vitest, Playwright, GitHub Actions*

Vortex — Enterprise LLM Gateway

Role: Solo Author — Full Stack Developer · **Type:** Personal Open-Source Project

Repository: https://github.com/kouji-dev/vortex

Project context: Vortex is a **multi-tenant (SaaS) or self-hostable, OpenAI/Anthropic-compatible LLM gateway** with enterprise governance — per-member budgets, role-based access control, audit trails and cost attribution — giving organisations a single, governed entry point to large language models.

- Architected a **pnpm monorepo** (feature-per-folder, hexagonal-lite) with a **Hono** API + gateway, two **Angular 22 (zoneless)** consoles — tenant and platform super-admin — and shared `core`, `db`, `shared`, `sdk` and `cli` packages
- Built an **OpenAI/Anthropic-compatible gateway** with a **provider registry**, routing all usage through a single governed endpoint with drop-in model substitution
- Implemented **enterprise governance** — **per-member budgets**, **RBAC**, audit logging and **cost attribution** per acting user / app — enforced across tenants
- Designed a **multi-tenant data layer** on **PostgreSQL + Drizzle** with **row-level security (RLS)** for tenant isolation, and **Redis** for caching and real-time usage counters
- Integrated **Stripe billing** (SaaS mode) and **Zod-validated DTOs** shared across apps, and shipped a **Vortex CLI** plus a thin **SDK** with acting-user / acting-app helpers
- Delivered the tenant and platform consoles with **@kouji-ui** (my own design system) and **Angular SSR**, covered by **Playwright** end-to-end tests and Docker-based local infrastructure

Technical Environment: *Angular 22 (zoneless), TypeScript, Hono, Node.js, PostgreSQL, Drizzle ORM (RLS), Redis, Zod, Stripe, @kouji-ui, Angular SSR, Playwright, Docker, pnpm, Render*

Kouji UI — Angular Design System & Component Library

Role: Solo Author & Maintainer · **Type:** Personal Open-Source Library

Repository: https://github.com/kouji-dev/kouji-ui

Project context: An open-source **Angular 21** design system providing **60+ accessible, themable UI primitives** covering inputs, overlays, navigation, data display, feedback and layout. Maintained as a single-author library to demonstrate end-to-end *frontend platform engineering* — monorepo discipline, publishable package boundaries, accessibility-by-default and rigorous release management.

- Architected a **pnpm + Turborepo monorepo** with three publishable packages — `components`, `core`, `themes` — plus a dedicated documentation app, sharing tooling, lint rules and a Tailwind v4 foundation
- Authored **60+ standalone Angular 21 components** (calendar, combobox, command palette, date/time pickers, dialog, drawer, file upload, form, OTP input, popover, stepper, toast, tooltip, tree-select, …) designed for tree-shakability and signal-based change detection
- Built a **decoupled theming layer** (`@kouji-ui/themes`) driven by design tokens, enabling theme switching, dark mode and per-product overrides without forking components
- Set up a **strict quality pipeline**: **Vitest** unit tests, **Playwright** end-to-end suites, automated **accessibility audits** (axe-based) producing JSON + HTML reports under `reports/a11y`, plus ESLint, Prettier and Husky pre-commit hooks
- Established **release engineering** with **Changesets** for semantic versioning and changelog generation across the three packages, wired into GitHub Actions workflows
- Wrote contributor documentation and **AI-collaboration guidelines** (`CLAUDE.md`, `RULES.md`) so the library can be extended consistently by both humans and AI coding agents

Technical Environment: *Angular 21, TypeScript, Angular CLI, pnpm workspaces, Turborepo, Vitest, Playwright, axe-core, ESLint, Prettier, Husky, Changesets, Tailwind CSS, GitHub Actions*

Pages — Unified Project Management & Documentation Platform

Role: Full Stack Developer · **Type:** Personal Side Project

Repository: https://github.com/kouji-dev/pages

Project context: Pages is an ambitious full-stack platform designed to **replace the traditional Jira + Confluence stack** with a single, cohesive solution. The goal is to eliminate context switching by unifying **issue tracking, sprint management, and technical documentation** into one modern, developer-friendly platform, while laying the foundation for **AI-enhanced workflows**.

- Established a **scalable monorepo architecture** using **Domain-Driven Design (DDD)** principles across frontend and backend
- Delivered a **highly performant, reactive frontend** with **Angular 21** and **Angular Signals** with reusable, responsive components
- Engineered a **production-ready REST API** with **FastAPI (Python 3.12)**, integrated with **PostgreSQL** via **SQLAlchemy 2.0** and **Redis**
- Implemented **end-to-end project management workflows** including sprint planning, issue tracking, Kanban boards, and backlog management with **real-time collaboration** via WebSockets
- Enabled **rich documentation authoring** through **Lexical editor** integration and ensured code quality with comprehensive test coverage (unit, integration, functional) and automated CI/CD workflows

Technical Environment: *Angular 21, TypeScript 5.7+, Python 3.12, FastAPI, PostgreSQL 17, Redis 8, SQLAlchemy 2.0, Lexical, Tailwind CSS 4, Docker, Docker Compose, Poetry, pnpm, WebSockets, JWT, Pydantic, Alembic, Git, GitHub Actions*

GamerGG — Gaming Service Platform

Role: Full Stack Developer · **Type:** Personal Side Project

Repository: https://github.com/kouji-dev/GamerGG

Project context: GamerGG is a **full-stack gaming service marketplace** designed to connect players with professional boosters and coaches. The platform enables gamers to purchase **rank boosting, coaching services, and gaming packages** while providing boosters with tools to manage orders, track performance, and engage with clients through integrated **Discord** communication.

- Architected a **Turborepo monorepo** structure enabling shared components and utilities across multiple Next.js applications with optimized build caching
- Built a **modern, responsive web application** with **Next.js 13**, **TypeScript**, and **Tailwind CSS**, implementing server-side rendering and API routes
- Designed and implemented **authentication and authorization** using **NextAuth.js** with OAuth providers and secure session management
- Developed a **robust database layer** using **Prisma ORM** with **PostgreSQL**, implementing migrations, seeding, and type-safe queries
- Created a **comprehensive order management system** with real-time status tracking, active boost monitoring, and order history for both clients and boosters
- Integrated **Discord server** connectivity for seamless communication between clients and boosters within the platform
- Implemented a **user analytics dashboard** featuring top boosters leaderboard, user spending tracking, and performance metrics
- Established a **reusable UI component library** shared across the monorepo with a consistent design system and TypeScript types

Technical Environment: *Next.js 13, React 18, TypeScript, Prisma, PostgreSQL, NextAuth.js, Tailwind CSS, Turborepo, Yarn Workspaces, OAuth, Discord API, Git, GitHub Actions*

Acta — Accounting Firm Project Management Platform

Role: Full Stack Developer · **Type:** Personal Side Project

Project context: Acta is a **comprehensive project management platform** designed specifically for accounting firms to streamline **cross-functional production tracking** across multiple teams. The platform enables real-time task planning, time tracking, and ticketing for complex accounting workflows, managing hierarchical structures of **portfolios, files, missions, and tasks** with progressive validation workflows and multi-dimensional progress visualization.

- Architected a **full-stack monorepo** with **Angular 21** frontend and **Express.js** backend, implementing **role-based access control (RBAC)** with granular permissions
- Designed a **complex hierarchical data model** using **Prisma ORM** with **PostgreSQL**, supporting portfolios → files → missions → tasks relationships with progressive validation workflows
- Built **multi-dimensional visualization features** including calendar views, progress tracking by collaborator or file, and time tracking dashboards with daily/weekly/monthly/yearly aggregations
- Developed **real-time task management** with status workflows, a due-date change-request system with approval workflows, and automatic task assignment based on role-based folder assignments
- Implemented a **comprehensive time tracking system** with detailed reporting, enabling analysis by collaborator, file, task, or mission with flexible filtering and aggregation
- Created **responsive, data-rich UI components** using **TanStack Table** for complex data grids, **Luxon** for date/time handling, and **Tailwind CSS** for modern, accessible design
- Established **robust authentication and authorization** using **JWT** and **Passport.js** with role-based route guards and permission-based feature access

Technical Environment: *Angular 21, TypeScript, Express.js, Prisma, PostgreSQL, JWT, Passport.js, TanStack Table, Luxon, Tailwind CSS, Docker, pnpm*

Chartify — Figma Plugin

Role: Plugin Developer · **Type:** Figma Community Plugin

Link: https://www.figma.com/community/plugin/1361309122482309141/chartify

Project context: Chartify is a **Figma community plugin** designed to streamline the creation of data visualizations and charts directly within Figma's design environment. The plugin enables designers to quickly generate various chart types, customize data visualizations, and integrate charts seamlessly into their design workflows without leaving Figma.

- Developed a **Figma plugin** using TypeScript and the Figma Plugin API to extend design-tool capabilities
- Implemented **chart generation functionality** supporting multiple chart types and customizable data visualization options
- Created an **intuitive user interface** within Figma's plugin environment for easy chart configuration and customization
- Ensured **seamless integration** with Figma's design system, allowing charts to be easily incorporated into design files
- Published and maintained the plugin in the **Figma Community**, making it accessible to designers worldwide

Technical Environment: *Angular, TypeScript, Taiga UI, Figma Plugin API, Figma Community*

EDUCATION

State Engineering Diploma in Computer Science
National School of Applied Sciences of Tangier — 2019

LANGUAGES

- Arabic — Native
- French — Fluent
- English — Professional