MUHAMMAD JEHANZAIB

Senior AI Engineer & Solutions Architect

Shipping production-grade AI systems across agentic automation, applied LLMs, computer vision, and medical AI.

mjehanzaib99@gmail.com | linkedin.com/in/muhammad-jehanzaib-4036b5195 | github.com/mjehanzaib999 | Portfolio: mjehanzaib-ai-portfolio.up.railway.app

SUMMARY

Senior AI Engineer and Solutions Architect with 5+ years shipping production AI systems, specializing in agentic LLM platforms, RAG, LLM fine-tuning and optimized inference, real-time voice, computer vision, and edge deployment. Distinguished by owning products end to end, from architecture and agent design through backend, frontend, deployment, and production hardening. Shipped 25+ production systems across tax automation, conversational AI with real-time voice, live AI SaaS products, MCP agent tooling, open-source LLM compression, and edge healthcare robotics. Published twice in Medical Image Analysis (Impact Factor 10.9). Bachelor of Science in Computer Science.

SKILLS & EXPERTISE

LLM Engineering: LangGraph · LangChain · LlamaIndex · LangSmith · Anthropic and OpenAI SDKs · OpenRouter · MCP · multi-agent orchestration · human-in-the-loop workflows · tool-calling · RAG · conversational memory · LLM compression (pruning, quantization, LoRA)

Voice & Real-Time: LiveKit Agents · speech-to-speech models (OpenAI Realtime, Gemini Live, Amazon Nova Sonic) · Deepgram STT · Cartesia TTS · ElevenLabs

ML & Computer Vision: PyTorch · TensorFlow · Hugging Face · scikit-learn · TensorRT · MMAction2 · MediaPipe · OpenCV · DeepFace

Languages & Frameworks: Python · FastAPI · Flask · Next.js · React · TypeScript · SQL / PostgreSQL · Streamlit

Cloud, MLOps & Infra: AWS · Azure · GCP · Docker · Kubernetes (K3s) · Ray · Celery · Redis · Railway · Vercel · Supabase · AWS SageMaker · GitHub Actions · CI/CD

Edge & Robotics: NVIDIA Jetson · Jetson Xavier · ROS · DeepStream · Azure Kinect · TFLite · ONNX

WORK EXPERIENCE

Full-Stack AI Engineer & Architect

Jan 2026 – Present

Ventive

Lead engineer on a **five-service platform** that automates IRC §41 R&D tax-credit qualification for a U.S. tax-advisory firm: an LLM contract-analysis engine, a FastAPI gateway, a React admin console, a tax-form extraction engine, and a §41 calculation and workpaper engine. Responsible for security, concurrency, and data-integrity hardening across all five services, including client data isolation.

- Architected the contract-analysis engine (Python, LangGraph, Celery, Redis) as a **multi-agentic workflow** in which specialized stages (triage, rights assessment, contingency, jurisdiction, adversarial review, IRS counter-argument, final determination) each evaluate one aspect of a contract against IRC §41 and §6662 standards. Each stage receives only the clauses relevant to it, and independent stages execute in parallel to reduce analysis time.
- Converted the model's raw confidence output into an **auditable §6662 Tax Position score** suitable for reviewer defense, with an automated approval gate that clears unambiguous cases and routes the remainder for human review.
- Built a **streaming multi-LLM chat interface** for reviewers, supporting model switching mid-analysis, clean cancellation of long-running jobs, and safe correction of results with protection against prompt injection.

- Designed **server-authoritative resumable sessions** so that long-running analyses persist across page reloads, navigation, and tab switches, with automatic repair of corrupted state and strict isolation between clients.
- Delivered a **multi-tenant automation platform** in which system events trigger configurable actions (for example, provisioning Google Drive folder structures) through a reliable event pipeline, with a no-code workflow editor and dashboards for runs and events.
- Built a **tax-form extraction engine** (AWS Textract with Claude post-processing) covering **29 IRS form types** (W-2/W-3, 1099, K-1, 1120/1120-S/1065, 6765, 2553, state forms), with per-form document segmentation and LLM-based form classification.

AI Engineer (Contract)

Oct 2025 – Mar 2026

Omvi AI · Watch demo ↗

Designed and built end to end "William," a multi-tenant agentic AI SaaS with real-time voice for a telecom client (World Mobile): users talk or type to an assistant that answers from the company knowledge base, acts in email and calendar, remembers earlier conversations, and turns a voice call into a project that a team of AI agents completes live.

- Architected the platform as **six services around one agent brain**: a **FastAPI gateway** through which every chat and voice request passes (Redis-backed sessions and history, streamed responses, **LangGraph onboarding flows** with pauses for user input), a **standalone LangGraph graph server** so the agent ships independently of the API, a LiveKit voice worker, a pgvector RAG service, a multi-agent orchestrator, and a Next.js studio.
- Built the **LangGraph agent**: it classifies each message, directs email and calendar requests to dedicated orchestrators, and routes all other requests to **knowledge-base search, cross-session memory recall** (with safeguards against fabricated recall), or a **tool-enabled responder** (knowledge search, web search, time, task creation), streaming tokens as they are generated.
- Built a **multi-agent orchestrator** that runs every task extracted from a call through **five specialised agents** (research, plan, build, review, write-up), with live WebSocket progress, per-agent output and token cost, instruction-driven re-runs, and downloadable reports.
- Built the **real-time voice agent on LiveKit** (Deepgram speech-to-text, Cartesia text-to-speech, Silero voice activity detection, noise cancellation) on the same agent core as chat, and **patched the LiveKit LangChain adapter** so that the agent, rather than the plugin, governs spoken output. Added a fast retrieval route with an instant spoken filler and warm workers so the agent joins a call in about 10 seconds; each call's transcript becomes a session of linked tasks.
- Built the **knowledge and model layer**: pgvector RAG with neighbouring-passage context and **multilingual retrieval**, every model routed through OpenRouter with automatic fallbacks, and token streaming fixed for LangGraph on Python 3.10, cutting the first reply from **35 seconds to under 10**.
- Designed and implemented **multi-tenancy**: organization-level data separation in Redis, role-based permissions, and tool access restricted to each user's entitlements. Authored the architecture plans adopted by the team for serving multiple clients from a single codebase.
- Shipped to GCP through GitHub Actions and later as **8 services on Railway** with auto-deploy from main and Playwright-verified releases, with per-provider token and latency tracking.

AI Solution Architect & Researcher (Contract, Founding Engineering Team)

Feb 2025 – Jan 2026

Optiml

Founding engineer on OptiML, an open-source framework for LLM compression and fine-tuning that produces smaller, lower-cost models for production. Responsible for architecture, CI/CD, release management, and a team of 4.

- Designed a single **config-driven API** covering **pruning (magnitude, Wanda, SparseGPT), quantization, LoRA fine-tuning, and MLA**, including hardware-friendly 2:4 structured sparsity, with integrated lm-eval benchmarking.

- Scaled training and compression across **multi-GPU NVIDIA and Intel environments** with PyTorch DDP.
- Built the GPU infrastructure: cost-capped Vast.ai GPUs provisioned into a **K3s cluster** over Tailscale, Ray on AWS EC2, a FastAPI job API with per-user quotas, and **self-hosted inference** through Text Generation Inference.
- Implemented AWS SageMaker JumpStart pipelines and GitHub Actions CI/CD; managed sprints and releases.

Senior Machine Learning Engineer

Feb 2021 – Feb 2025

Xavor Corp USA

Built the AI modules of "Ruyi," a robot that supports the recovery of elderly and post-surgery patients using personalized longitudinal data, together with a medical data bank for clinicians integrated with social-media intelligence.

- Developed the **ADL (Activities of Daily Living) module**, which recognizes patient activities and measures the effort and duration of each movement, running at **30 FPS on the device**.
- Built an **audio module recognizing 9 health-critical sounds**, and a vision pipeline that converts robot sensor data into a structured patient record for recovery planning.
- Co-developed a **gait analysis module tracking 15 clinical parameters**, from step length and knee flexion to cadence.
- Deployed across **NVIDIA Xavier edge devices**, Microsoft Azure, and AWS.
- Built data scrapers, cleaning routines, and a **generative-AI social-media intelligence** layer supporting medical-data explainability.

PUBLICATIONS

FFPE++, *Medical Image Analysis (Impact Factor 10.9)*
Unpaired image-to-image translation with mixed channel-spatial attention for histopathology artifact correction.
doi.org/10.1016/j.media.2023.102992

MediSeg & MediGan, *Medical Image Analysis (Impact Factor 10.9)*
Novel attention-based segmentation and semantically controlled synthetic generation for histopathology datasets.
doi.org/10.1016/j.media.2024.103344

FEATURED WORK

Forensic Lease Audit Engine
Agentic system that ingests leases, amendments, prior true-ups, and invoices, extracts and interprets the governing clauses, and validates each line item against them and against external data sources (weather records, utility benchmarks). A LangGraph workflow runs clause extraction, cross-checks, and year-over-year anomaly detection as separate stages, with vector retrieval over the lease corpus to locate the relevant clause for every charge. Each finding carries a clause-cited justification, and auditors can query, challenge, and override findings through a conversational agent with access to the same retrieval tools. Automated approximately 90% of the manual audit process.
LangGraph · OpenAI · Vector retrieval · React · Python · External APIs

Biologics Intelligence: Citation-Verified Research Assistant (Watch demo ↗)
Sole engineer, from architecture to production handover, of a competitive-intelligence platform for a medical-device consultancy. Research questions that took analysts days of reading across FDA records, clinical trials, journals, and SEC filings now return in **about 2 minutes for about $1 in model spend**, with every sentence tied to a source passage a reviewer can open. Built the **ingestion pipeline** (scheduled syncs from five public sources, PDF parsing, entity resolution over **33,800 documents, 668,000 passages, and 711 tracked products**), **hybrid retrieval** (vector, keyword, and sentence-window lanes merged by rank fusion over a pgvectorscale filtered index), and a **five-stage LangGraph workflow**: planner, retrieval, a judge model that reads every candidate passage, a writer, and a checker that **verifies each sentence against its citation** before display, refusing when the evidence is not there. Cut a hard question from **6 minutes and $4 to 2 minutes and $1** through retrieval and judge optimisation. On the client acceptance set, **every regulatory number was correct** against the FDA record. Shipped with live progress, audit trail, per-stage model controls, cost reporting, an admin console, scheduled backups, and full operations and user documentation.
Python · FastAPI · LangGraph · PostgreSQL + pgvector + pgvectorscale · Celery + Redis · Next.js · TypeScript · OpenRouter (GPT-5.5, gpt-oss) · Railway

AI Onboarding Ops: Employee and Client Onboarding as an MCP Server (onboarding.kognio.ai)
A web console and a **Model Context Protocol (MCP) server** from one codebase, which any AI client can call with a personal token. One instruction starts an employee or client onboarding (person record, tasks, documents, accounts); tasks that need paperwork stay closed until the document is accepted, and policy questions are answered only from the versioned handbook with a citation per claim. **25 narrow tools sit behind one guard** that runs the role check, idempotent replay, the handler, and the audit write on success and on failure, so no front door can change data without a record; restricted personal data never reaches chat or MCP.
Python · FastAPI · PostgreSQL · MCP (JSON-RPC 2.0) · OpenRouter (Claude Sonnet 4.5) · React 18 · TypeScript · Railway

PropertyAI CRM: AI Property Management Platform (propertyai.kognio.ai)
AI tenant-communication layer on top of Buildium for property managers. Inbound tenant SMS (Vonage) is classified and handled autonomously by a Claude-based agent: rent follow-ups, maintenance intake with automatic work-order creation through the Buildium REST API, lease-renewal outreach with ReportLab PDF generation, and escalation to staff when confidence is low, with per-tenant conversation memory persisted in PostgreSQL. APScheduler drives the follow-up and renewal cadences. Includes a portfolio copilot grounded in the manager's Buildium data, a Buildium-mirrored dashboard, a WhatsApp-style inbox, and JWT-based role-based administration.
React 18 · TypeScript · FastAPI · PostgreSQL · OpenRouter (Claude Sonnet 4.5) · Vonage SMS · Buildium REST API · APScheduler · Railway

Commerce IQ: Commerce Intelligence Platform (commerce.kognio.ai)
Decision-support platform for Amazon-led D2C brands. Scheduled ingestion jobs (APScheduler) consolidate orders, inventory, advertising, and returns data into a canonical PostgreSQL warehouse, where deterministic scoring rules and AI workflows run against it. Findings surface through a React dashboard, a tool-calling chat agent that queries the warehouse directly, Redis-backed real-time alerts, and a daily Telegram digest. Includes lead-time-aware reorder recommendations, a true-profit calculator that nets fees and returns, Buy Box monitoring, and returns and refunds tracking, with multi-model routing through OpenRouter.
React 18 · TypeScript · FastAPI · SQLAlchemy 2.0 (async) · PostgreSQL 16 · Redis 7 · OpenRouter (multi-model) · APScheduler · python-telegram-bot · Railway

Cure Health: Unified Digital Health Platform (curehealth.kognio.ai)
Multi-portal digital health platform covering the full care loop across six role-scoped dashboards: booking, visit, e-prescription, pharmacy dispensing, insurance adjudication, and financing. Built on a FastAPI backend with async SQLAlchemy and Alembic migrations over PostgreSQL, exposing 46 API endpoints to a Next.js 16 front end with 24 role-scoped screens. AI clinical support includes symptom triage, a streaming care navigator, and clinical-note drafting by a tool-calling agent, all served through OpenRouter (Claude), together with in-browser WebRTC telehealth.
Next.js 16 · React 19 · TypeScript · FastAPI · async SQLAlchemy · Alembic · PostgreSQL · WebRTC · OpenRouter (Claude) · Railway