NOUMAN AHMAD — Full Stack AI / ML Engineer

Last updated: November 2026

Lahore, Pakistan — nouman.ahmad.8848@gmail.com GitHub

Summary

AI/ML Engineer with hands-on experience building LLM, agentic RAG, multi-agent, and multimodal AI systems across desktop, cloud, and multi-tenant environments, with evaluation-driven decisions using LLM-as-judge harnesses. Experienced in retrieval pipelines, document/audio/video ingestion, vector databases, embedding workflows, reranking, prompt orchestration, and source-grounded LLM responses. Skilled in Python, FastAPI, Celery, LlamaIndex, LangChain, Qdrant, OpenAI/Azure OpenAI, Gemini, Anthropic, Electron, React, and Docker. Familiar with scalable AI backend patterns including asynchronous workers, caching, usage tracking, billing-related workflows, offline/local model support, and AWS deployment components such as ECS Fargate, RDS, Redis, CloudWatch, and CI/CD pipelines.

EXPERIENCE

Machine Learning Engineer — LevitateData, Remote *Aug 2026 – Present*

- Built a multi-agent "team" feature for an enterprise agentic RAG platform, where a supervisor agent delegates questions to specialist agents with their own knowledge bases and merges cited answers, with parallel delegation, retries, time limits, and audit logging.
- Extended an evaluation harness with LLM-as-judge grading, bootstrap confidence intervals, and pre-defined pass criteria to compare multi-agent and single-agent setups across 13 tests (team won 3, tie 6, single won 4); teams were not made the default, as a standard supervisor lost and teams won only with a strong supervisor that rewrites answers or an ask-every-specialist rule.
- Reduced multi-agent response time by ∼33% through parallel specialist queries, and designed a routing rule that improved fact recall by ∼5 points on questions with ambiguous terms.
- Measured cost/quality trade-offs across model tiers (∼35–60% lower cost per answer vs. ∼5-point recall loss) and used diagnostic tooling to trace quality gaps to model choice.
- Built multi-tenant authentication and access control on AWS Cognito, including role-based permissions, multi-account membership, session revocation, token refresh, invitations, email verification, and Google sign-in, using FastAPI, PostgreSQL, AWS CDK, and Lambda.
- Built scoped API keys with tier/project limits, expiry, and spend alerts, per-account and per-IP rate limiting with Redis, and closed cross-tenant data-access gaps found in review.
- Built usage-based billing for agent runs using a transactional outbox pattern with Celery, Redis, and PostgreSQL, plus transactional email on AWS SES with per-tenant send limits and audit trails.
- Wrote an end-to-end test kit running against real AWS Cognito, SES, and Postgres alongside unit and integration tests.

AI/ML Engineer — WebifyAI, Remote *May 2024 – Aug 2026*

- Worked on AI ingestion and retrieval systems for desktop and cloud applications using Python, FastAPI, LlamaIndex, Celery, Qdrant, OpenAI/Azure OpenAI, Gemini, Anthropic, and AWS.
- Contributed to a cross-platform AI desktop application with a FastAPI backend and Electron/React frontend. Worked on extraction pipelines for PDFs, DOCX files, images with OCR, websites, YouTube videos, and local files, with IPC-based communication between frontend and backend services.
- Implemented and improved multi-format document ingestion workflows, including semantic/basic chunking, metadata extraction, Pydantic schema validation, vector collection handling, embedding consistency checks, caching, reranking, and local model support for offline inference.
- Contributed to building and maintaining a multi-tenant RAG platform with configurable ingestion, chat/query RAG, schema enrichment, audio ingestion, video ingestion, usage tracking, and provider-based configuration through an internal RAG package.

- Worked on production RAG workflows covering document parsing, chunk-level enrichment, embedding generation, Qdrant storage, filtered retrieval, optional reranking, prompt construction, LLM responses, source references, conversation history, and webhook callbacks.
- Added audio and video ingestion features using Deepgram and Vertex Gemini, including URL-based ingestion, GCS-backed video processing, long-context usage tracking, billing-related integration, and automated tests.
- Built and debugged asynchronous Celery workers for ingestion, chat, and billing-related pipelines, including queue routing, retries, status reporting, usage outbox processing, and worker-level logging.
- Worked with AWS-based deployment workflows involving CDK, Docker, ECS Fargate, ALB, RDS Postgres, ElastiCache Redis, ECR, CloudWatch, Secrets Manager, SSM, and GitHub Actions-based build/deploy pipelines.

AI/ML Engineer Intern — Netsol Technologies, Pakistan *Jan 2025 – Mar 2025*

- Built a **CV information extraction pipeline** for unstructured resumes using preprocessing, prompt engineering, Gemini API, custom algorithms, and JSON schema-based structured extraction.
- Developed algorithms to compute total professional experience and calculate durations across varied time-based datasets.
- Deployed interactive **Streamlit** applications for model visualization and strengthened deep learning fundamentals by implementing **ANNs** from scratch.

SKILLS

- ✓ **AI/ML & LLM Systems**: RAG, agentic workflows, LLM application development, prompt engineering, embeddings, reranking, semantic search, grounded generation, conversational memory, local inference, LLM-as-judge evaluation, bootstrap confidence intervals
- ✓ **Agents & Retrieval Infrastructure**: LlamaIndex, AI agents, multi-agent orchestration (supervisor/specialist), tool calling, multi-step workflows, Qdrant, vector databases, filtered retrieval, chunking, metadata enrichment, source referencing
- ✓ **Production Backend Development**: Python, FastAPI, Pydantic, REST APIs, async processing, multi-tenant systems, RBAC/auth, scoped API keys, rate limiting, transactional outbox, webhooks, provider configuration
- ✓ **Data Ingestion & Document AI**: PDF/DOCX parsing, OCR, web, YouTube, audio/video ingestion, schema validation, metadata extraction
- ✓ **LLM Providers & AI APIs**: OpenAI, Azure OpenAI, Gemini, Anthropic, Vertex AI, Deepgram
- ✓ **MLOps & Production Workflows**: Celery, queue routing, retries, worker logging, status tracking, caching, automated tests, usage tracking, billing pipelines
- ✓ **Cloud & DevOps**: AWS CDK, Cognito, SES, Lambda, Docker, ECS Fargate, ALB, RDS Postgres, ElastiCache Redis, ECR, CloudWatch, Secrets Manager, SSM, GitHub Actions
- ✓ **Databases & Storage**: PostgreSQL, Redis, Qdrant, AWS S3, Google Cloud Storage
- ✓ **AI-Assisted Development**: Claude Code (sub-agents, hooks, skills), agentic coding workflows, token/cost tracking
- ✓ **Frontend & AI Applications**: Electron, React, Streamlit, cross-platform desktop apps, AI model visualization

Personal AI Engineering Projects

AliceSpeaks – In Progress

- Building a **local-first AI desktop voice assistant** with PySide6, local LLMs, realtime speech input, and human-confirmed actions.
- Implemented modular **STT, LLM, TTS, audio, and tool** providers, including Whisper, Distil-Whisper ROCm/PyTorch, Kokoro, F5-TTS, and local sidecar services.

- Designing the assistant toward a **local animated persona frame**, where an image/video-based character can speak, react, and perform actions based on the active conversation or task.
- Prioritizing fully local execution to reduce data-leak risk, with offline-capable models, local audio processing, and confirmation-gated desktop automation.

LogsMate – Local LLM Log Analysis System

- Built a post-event log analysis system using **RAG**, **BM25**, **FlagEmbedding reranking**, and **ChromaDB** to generate structured incident summaries from raw logs.
- Integrated **Ollama** with **Llama 3.2 3B** for local inference and private/offline analysis.
- Designed retrieval and report-generation workflows for identifying errors, timelines, root-cause signals, and recommended fixes.

NPC-AI – Persona-Grounded Multi-Agent Dialogue Simulator

- Built an LLM-based dialogue simulation system where fictional NPCs interact using character cards, world context, scene details, memory, and personality constraints.
- Implemented structured prompts for role consistency, contextual awareness, long-running conversations, and character-specific response behavior.
- Designed the system to generate dynamic, unscripted conversations that preserve each NPC's personality, backstory, and relationship to the fictional world.

Game Reverse-Engineering & Modding Toolkit – Agentic AI-Assisted Development

- Designed and directed an **agentic coding workflow** in **Claude Code** (lead agent, sub-agents for reading/building/review, rule-enforcing hooks, per-topic state files) across 5,000+ commits, measuring token usage per session to reduce cost.
- Reverse-engineered undocumented **binary data formats** (archives, data tables, visual-script graphs, behaviour trees) and recovered 1,895 type names by matching **xxHash64** hashes against the executable.
- Built a Python toolkit indexing all game data into a **SQLite** catalog with incremental rebuilds, answering "where is X used" across thousands of files in seconds, backed by ∼2,800 automated tests and mock runs of game logic via **lupa**.
- Shipped an open-source **C#/WPF** mod manager and mod packs on Nexus Mods with an automated release flow, sha256-verified installs, and one-step undo.

Legacy Game Engine Graphics Upgrade – Open-Source C++/OpenGL

- Added modern rendering to a 1999 open-source engine (Open1560): dynamic **shadow maps** with a day cycle, per-pixel lighting, planar reflections, and bloom, each behind a runtime flag.
- Ported legacy **x86 assembly to C++** using a "twin" harness that runs old and new code side by side, plus frame pixel-diff and replay checks.
- Wrote byte-exact readers/writers for legacy mesh formats with headless **Blender** round-trip tests.

EDUCATION

[2021 – 2025] BS (Computer Science) - FAST NUCES, Lahore

CGPA: 3.1 / 4.0