Hitendra Kawale

+44 7818916926 | hituhitesh303@gmail.com | linkedin.com/in/hitendra-kawale | github.com/HitendraKawale

Education

University of Surrey — Guildford, UK
MSc Artificial Intelligence, Merit — *Feb 2025 – Feb 2026*

Savitribai Phule Pune University — Pune, Maharashtra
BE Computer Engineering with Honours in AI/ML, CGPA: 8.61/10 — *Sep 2020 – May 2024*

Skills

Languages: Python, SQL, Bash, Lua (Neovim plugins)
AI / ML: PyTorch, Hugging Face, LLMs, LangGraph, RAG, Sentence Transformers, UMAP, 3D Gaussian Splatting, Ollama
Vector Databases: Qdrant, pgvector, LanceDB, FAISS
Backend / APIs: FastAPI, Flask, REST APIs, PostgreSQL, Redis
Cloud / Storage: AWS EC2, AWS S3
DevOps / Observability: Docker, Docker Compose, GitHub Actions, CI/CD, Prometheus, Grafana
Frontend / Tools: Next.js, TypeScript, Playwright, Streamlit, Postman, Git, Neovim, Linux
Spoken Languages: English (fluent), Spanish (fluent), Mandarin (intermediate), Hindi & Marathi (native)

Experience

Perpetual Solutions LLP — Remote
Machine Learning Engineer Intern — *Jul 2024 – Dec 2024*

- Built an interview-screening assistant on Hugging Face transformer models (DistilBERT, RoBERTa) that scored candidate responses, triggered follow-up questions, and generated recruiter-facing summaries, validated across ∼20 simulated interviews.
- Designed a chunking and indexing pipeline over 300+ resumes (AWS S3 storage, PostgreSQL metadata), cutting document lookup to under 2 seconds per query.
- Lifted answer relevance from 68% to 84% on a 50-query test set by tuning chunk size, top-k retrieval, and embedding selection against recruiter feedback.
- Deployed services to AWS EC2 and built an edge-case Postman regression suite that cut manual regression time per release by ∼40%.

Tech: Flask, FastAPI, DistilBERT, RoBERTa, Hugging Face, PostgreSQL, AWS EC2, AWS S3, Streamlit, Postman

Projects

Mini OpenAI Platform

- Architected a microservices LLM platform (API gateway, RAG, embedding, and inference services) exposing OpenAI-compatible APIs, with request tracing, multi-key constant-time auth, rate limiting, and 16 Prometheus/Grafana panels covering latency, token economics, and answer quality.
- Built an embedding-based semantic cache that serves paraphrased queries without invoking the LLM, cutting response latency ∼80× (10.8s → 0.13s) on cache hits, with TTL expiry, corpus-change invalidation, and saved GPU-seconds exported as a Prometheus metric.
- Designed a difficulty-based model router that scores each prompt with an explainable heuristic and dispatches it to a small or large Ollama model with automatic tier failover; the difficulty score is returned in every API response for debuggability.
- Implemented a RAG evaluation quality gate in CI: a golden dataset graded on recall@k and MRR (plus LLM-judged answer faithfulness) that fails the build on retrieval regression — caught a repo-breaking packaging bug on its first run.
- Shipped a React frontend that surfaces platform internals per message (serving model, routing decision, source chunks with similarity scores), served via an nginx single-origin proxy so no internal service is exposed.

Tech: Python, FastAPI, React, nginx, Docker Compose, Qdrant, Ollama, Prometheus, Grafana, GitHub Actions

RAG Chunk Visualizer — GitHub | Live demo

- Built an interactive RAG debugging tool that visualizes every stage of the retrieval pipeline — document chunking, embedding generation, 2D embedding maps (PCA/UMAP), vector search, and grounded prompt construction — making normally opaque retrieval behavior inspectable point-by-point.
- Designed a unified LLM abstraction layer supporting 5 providers (Anthropic Claude, OpenAI GPT, Google Gemini, xAI Grok, local Ollama) behind a single interface, with a provider registry, per-provider typed error mapping, and runtime model/provider switching from the UI.
- Engineered a local-first embedding pipeline (sentence-transformers + LanceDB) so document content never leaves the machine — only the final generation step optionally calls a cloud API; API keys resolve via a session-only → environment-variable chain and are never written to disk or logs.
- Shipped with production practices — 59 pytest unit tests, ruff linting, validated environment-driven configuration, structured logging — and deployed to Hugging Face Spaces with a slim CPU-only PyTorch Docker image, several GB smaller than default torch installs.

Tech: Python, Streamlit, LanceDB, sentence-transformers, Docker, Hugging Face Spaces

Scribble-Guided 3D Scene Segmentation

- Extended the SegAnyGaussian (SAGA) pipeline with a scribble-based interactive segmentation framework; designed a feature embedding pipeline to map user scribbles into 3D Gaussian space for direct scene manipulation.
- Implemented Context-Aware Filtering (CAF) using k-NN graphs and radius-based connectivity to suppress floaters and fragmented regions; introduced iterative add/remove/overwrite refinement modes with ∼1s scribble processing and ∼3–4s filtering latency.
- Evaluated on 360V2 and NeRF-LLFF datasets; achieved a 1–2% precision gain over the SAGA baseline with a notable reduction in floater artifacts from CAF, while maintaining competitive IoU and Dice scores.

Tech: Python, PyTorch, CUDA, OpenCV, kNN, 3D Gaussian Splatting (SAGA)

Chat2Study — GitHub

- Built a full-stack RAG application (Next.js 15/React 19, FastAPI, PostgreSQL + pgvector, MinIO, Docker) that converts long AI chat transcripts into searchable knowledge bases, AI-generated study notes, and interactive concept maps.
- Designed a 10-node LangGraph ingestion pipeline orchestrating Playwright browser capture, artifact persistence (HTML/text/screenshot/PDF) to S3-compatible storage, text chunking, vector embedding, and conditional LLM note generation based on a content-complexity heuristic.
- Re-architected ingestion from synchronous request handling to an async job model (202 + job polling), cutting API response time for ingestion requests from 30–90s to ∼10ms; semantic search and grounded Q&A over pgvector embeddings run at ∼50ms retrieval latency.
- Built a provider-agnostic LLM factory over LangChain (Google Gemini, OpenAI, Anthropic, Ollama) for both chat and embedding models — switching providers is a single environment variable; kept retrieval correct across switches by recording the embedding provider per chunk and embedding queries with the same model.
- Hardened for production: JWT auth with per-user data scoping across all endpoints, environment-gated startup checks that refuse insecure secrets, structured logging, sanitized error responses, liveness/readiness probes, and GitHub Actions CI with containerized services and automatic Alembic migrations.

Tech: Python, TypeScript, Next.js 15, React 19, FastAPI, LangGraph, LangChain, Playwright, PostgreSQL, pgvector, MinIO, Ollama, Docker

Extracurricular

Google Developer Student Clubs (GDSC) Sep 2022 – May 2023

Volunteer — NBNSSOE, Pune

- Co-organised technical events and workshops for the GDSC chapter covering cloud, ML, and web development, coordinating logistics, speaker sessions, and student outreach across the college community.
- Contributed to building the club website, working on the frontend to keep event listings, announcements, and member information up to date.

Bhumi NGO Jul 2023 – Jan 2024

Volunteer Teacher — Pune, Maharashtra

- Taught computer literacy, MS Office, and introductory Python to 9 underprivileged students, designed exercises to make technical concepts accessible and engaging.
- Mentored students on problem-solving skills, fostering confidence and active classroom participation.