

Jaspreet Singh Nahal

AI/ML Engineer experienced in building production LLM pipelines, RAG systems, and autonomous agent architectures

Delhi, India | +91-9999604054 | jaspreetnahal100@gmail.com | linkedin.com/in/jaspreet-nahal
| github.com/jaspreet-nahal | jaspreetnahal.xyz

SUMMARY

AI/ML Engineer experienced in building production LLM pipelines, RAG systems, and autonomous agent architectures. Pretrained a 205M-parameter GPT Transformer from scratch in PyTorch. Deployed RAG over 10,000+ documents in production. Building Aria, a local-first autonomous digital employee using LangGraph and FastAPI.

TECHNICAL SKILLS

Languages: Python, Java, C++, JavaScript, TypeScript, SQL

Machine Learning & AI: PyTorch, TensorFlow, Hugging Face Transformers, SentenceTransformers, FAISS, LangChain, LangGraph, RAG, NLP, Instruction Fine-Tuning, Machine Learning, Deep Learning, Computer Vision, Prompt Engineering, LLM, OpenAI API, Gemini API, Anthropic API, Vector Databases

Data Science: NumPy, Pandas, scikit-learn, Matplotlib, HuggingFace Datasets, Tableau, Power BI, Data Visualization, Excel

Backend & DevOps: FastAPI, Flask, REST APIs, PostgreSQL, MongoDB, Git, Docker, GitHub Actions, Streamlit

Frontend: React.js, TypeScript, D3.js, Vite, TailwindCSS

EDUCATION

Dr. A.P.J. Abdul Kalam Technical University (AKTU, formerly Uttar Pradesh Technical University) 2023 – 2027

Bachelor of Technology in Computer Science Engineering (AI)

Uttar Pradesh, India

EXPERIENCE

Expedien eSolutions

Aug 2025 – Oct 2025

Machine Learning Intern

Noida, India

- Engineered a RAG-based legal query resolver using LangChain, FAISS, and Hugging Face Transformers, enabling accurate document-grounded responses over a corpus of 10,000+ legal documents
- Designed and optimized the document retrieval pipeline using vector embeddings and semantic search, achieving sub-2 second end-to-end response time with a Streamlit UI, improving query throughput by 40%
- Applied instruction fine-tuning techniques and prompt engineering to improve response quality and reduce hallucinations across the production pipeline

Freelance

Nov 2024 – Dec 2024

Algorithmic Trading Developer

Remote

- Built a live automated trading system with SMA Crossover strategy via Flask REST API and PostgreSQL; reduced execution latency by 30% through async processing and optimized SQL queries

PROJECTS

EverydayGPT

arxiv.org/abs/2606.11212

PyTorch, FAISS, RAG, NLP

Confidence-gated routing for efficient and safe hybrid GPT-RAG conversational QA

- Built a routing system that decides whether to answer queries via fast retrieval or full GPT generation, skipping generation for 85% of queries and reducing average latency by 6.3× while maintaining answer quality
- Pretrained a 205M-parameter GPT from scratch on 10B tokens; benchmarked against 8 baselines including LangChain RAG; system runs fully on consumer CPU under 2 GB RAM; published on arXiv

CodeAtlas

github.com/jaspreet-nahal/CodeAtlas

Python, FastAPI, LangChain, RAG, NetworkX, React, TypeScript

AI-powered code intelligence platform with RAG pipeline over GitHub repositories and ZIP archives

- Built an AI-powered code intelligence platform with a RAG pipeline over GitHub repositories and ZIP archives, enabling natural language Q&A, semantic concept extraction, and dependency graph generation across 8 programming languages
- Implemented domain-aware file classification and an adaptive learning layer personalizing code explanations by proficiency; deployed via async FastAPI with a React force-graph UI for real-time codebase exploration

Aria

LangGraph, FastAPI, Ollama, RAG, FAISS, React, Python

Local-first autonomous desktop agent converting natural language into structured OS-level task execution

- Building a local-first autonomous desktop agent that converts natural language into structured OS-level task execution across files, email, and browser with zero cloud dependency; architecting a multi-agent debate layer via LangGraph where competing sub-agents verify action correctness before irreversible execution
- Implementing a tiered safety model classifying actions as auto-execute, confirm, or reject; designing persistent memory for organizational context and task history with FastAPI backend, Ollama local inference, FAISS retrieval, and React frontend