# Aditya Anil Raut

San Jose, CA | araut1@csuchico.edu | +1 (669) 499-7554 | linkedin.com/in/adityaanilraut | github.com/adityaanilraut

## Education

**California State University, Chico** Master of Science in Computer Science Aug 2024 – Present
**University of Mumbai** Bachelor of Engineering in Computer Engineering Jun 2020

## Experience

**Tata Consultancy Services (TCS)** Apr 2021 – Oct 2023
System Engineer

- Developed a **web-scraping + similarity-search** pipeline: scraped multi-source content with **Scrapy**, handled JS rendering and **rate limits**, cleaned & chunked text, generated **embeddings** with **Sentence-Transformers**/**OpenAI**, and built a **FAISS** index with a retrieval API; achieved **95% precision** and **recall@K**.
- Built and shipped production services/APIs in Python on AWS Lambda added caching and containerized with **Docker**.
- Delivered **KPI analytics** end-to-end: modeled data in **SQL**, built stakeholder dashboards in **Tableau**, and defined **data contracts**; enabled **self-serve insights** that lifted target KPIs by **8–12%**.
- Architected and managed **distributed workloads** using **Docker** and **Kubernetes**, deploying and maintaining **scalable infrastructure** on **AWS** (**EC2**, **S3**, **EKS**).

## Skills

**Languages:** C++, Java, Python, PL/SQL, JavaScript, C, MATLAB, Swift.
**Technologies & Frameworks:** Git, FastAPI, React.js, REST APIs, Node.js, Bootstrap, HTML, CSS3, Tableau, Talend, Alteryx, AWS (SageMaker, EC2, ECS, S3), Docker, Kubernetes, PySpark, Power BI, Postman.
**Databases:** MySQL, MongoDB, Chroma, Firebase, PostgreSQL, Redis.
**Machine Learning:** CUDA, NumPy, Pandas, Transformers, Scikit-learn, Matplotlib, Seaborn, PyTorch, TensorFlow, NLTK, boto3, SciPy, BeautifulSoup, NLP, Deep Learning, LangFlow, Requests, Pillow, LangChain, Flask, FAISS.

## Projects

**Code Assistant CLI** github.com/adityaanilraut/homebrew-coderai

- Designed and shipped **CoderAI**, an **agentic CLI** with **multi-LLM backends** (**OpenAI** GPT-5* family + **LM Studio** local models) and **dynamic model switching**, enabling side-by-side **latency**/**cost**/**quality** comparisons in one session.
- Built a **tool-use agent** via **Model Context Protocol (MCP)**: **file I/O**, **terminal exec**, **Git ops**, **semantic** + **grep code search**, **web-docs lookup**, and lightweight **memory**—supporting end-to-end coding workflows from the terminal.
- Designed **pre/post-execution hooks**, **slash-command UX**, and **interactive**/**one-shot** modes to streamline developer workflows.

**Fine-Tuning Large Language Models (LLMs)** github.com/adityaanilraut/Finetuning-Google-Gemma2

- **Fine-tuned** an **LLM** to personalize output for a particular task or action.
- Optimized performance and reduced storage costs by **30%** using **LoRA**; **quantized** a 16-bit model to **4 bits** (**Gemma-2B**) for faster throughput and performance.

**Search Engine — RAG**

- Developed an **AI-powered search** tool using **Retrieval-Augmented Generation (RAG)** to process search engine results, extract key insights, and generate concise, context-aware summaries.
- Leveraged **cosine similarity** to rank retrieved documents based on **semantic relevance** to the query, improving precision in information retrieval and summary generation.

**Chess Engine** github.com/adityaanilraut/Chess-engine

- Built a chess engine using the **Minimax** algorithm with **alpha–beta pruning** to efficiently predict optimal moves by reducing the search space; **Flask** for backend and **JavaScript** for interactive UI.

## Awards & Hackathons

**Wefunder AI Hackathon — Context Router (Winner)** Link

- Developed **Context Router**, an intelligent **LLM-routing system** that dynamically analyzes user queries and selects the most suitable large language model based on **token length**, task complexity, and required **reasoning depth**.
- Won the **Pond Challenge** among 20+ teams; recognized for designing a novel **LLM orchestration** strategy that reduced infrastructure waste and introduced a modular, **cost-aware** deployment paradigm.
- Achieved up to 20% reduction in **API costs** by implementing prompt classification and **cost-aware model switching**.