Durvesh Vivek Patil

+91-8660994302 · durvesh.imp@gmail.com · linkedin.com/in/durvesh-patil-86216a2a8 · github.com/Durvesh717

Education

Vellore Institute of Technology — Vellore, India
B.Tech. in Computer Science and Engineering (CSBS); CGPA: 9.13/10 — *2023 – Present*
Geetanjali Olympiad High School (CBSE) — Bangalore, India
Class 12; Percentage: 91.6% — *2022*

Technical Skills

- **Languages:** Python, JavaScript, C/C++, Java
- **Backend & APIs:** FastAPI, Django REST Framework, Celery, SQLAlchemy 2.0 (Async), RESTful APIs, SSE, JWT Authentication, Argon2 Password Hashing, RBAC
- **Agentic Systems:** LangChain, LangGraph, Strands SDK, AWS Bedrock (Claude, Nova, Titan), AgentCore Runtime & Gateway, MCP Servers, RAG, Corrective RAG, Prompt Engineering, Hybrid Search (Dense + BM25), Cross-Encoder Reranking, RAGAS
- **Databases & Storage:** PostgreSQL (pgvector, RDS), ChromaDB, Redis, AWS S3
- **Cloud & DevOps:** AWS (Lambda, ECS Fargate, SQS, CloudFront, API Gateway, Cognito, CloudFormation), Terraform, Docker, GitHub Actions CI/CD
- **Tools:** Git, LangSmith, PyMuPDF, Streamlit, pytest, Playwright

Projects

JobSeek — GitHub
FastAPI, PostgreSQL (pgvector), AWS Bedrock, Docker
- Architected the backend for an AI-powered job matching and resume automation platform using FastAPI, supporting four distinct user roles (Applicant, Recruiter, Company Owner, Admin) with a React-based frontend.
- Designed an async, cookie-based JWT authentication system with Argon2 password hashing and role-based access control across the platform.
- Implemented an AI resume parsing and building pipeline using AWS Bedrock (Nova Micro/Pro, Titan Embeddings V2) to extract structured data from PDFs and programmatically generate ATS-friendly LaTeX resumes compiled server-side via pdflatex/tectonic.
- Engineered semantic job-matching using PostgreSQL with pgvector, performing cosine similarity search between resume and job description embeddings.
- Developed a streaming career-advisor chat assistant using AWS Bedrock (Nova Pro) with Server-Sent Events, and an automated fraud-detection system that scores and flags suspicious job postings.

PDF Q&A Bot (Agentic RAG) — GitHub
Python, LangChain, LangGraph, Streamlit
- Built a modular, agentic Retrieval-Augmented Generation (RAG) system implementing a self-correcting Corrective RAG (CRAG) loop using LangGraph, with dynamic query rewriting and live web-search fallback.
- Implemented multimodal PDF ingestion with PyMuPDF4LLM, extracting text, tables, and using a vision LLM to generate searchable descriptions of embedded images/charts.
- Designed a hybrid retrieval pipeline combining dense vector search and BM25 keyword search via Reciprocal Rank Fusion (RRF), followed by cross-encoder reranking (bge-reranker-base) to improve retrieval precision.
- Built LLM-based graders for retrieval relevance, hallucination detection, and answer usefulness to ensure grounded, high-quality responses with citations.
- Established a multi-cloud/model abstraction layer enabling runtime hot-swapping between Google Gemini, OpenAI GPT, and AWS Bedrock (Claude), and evaluated system quality using the RAGAS framework with LangSmith tracing.

Clyro — Live
React 19, Django REST Framework, PostgreSQL, AWS Bedrock, Terraform
- Built an AI-powered infrastructure provisioning platform that scans a connected GitHub repository and deploys a full AWS architecture through a 7-step guided wizard, using Django REST Framework, PostgreSQL, and Terraform.
- Architected a multi-agent AI system with 3 specialized agents (RepoRecon, Reasoning, IacArchitect) built on the Strands SDK and Amazon Bedrock AgentCore, enforcing strict trust boundaries between the code-scanning agent and the AWS-provisioning engine.
- Replaced LLM-based CloudFormation generation with a deterministic Python generator, reducing IaC generation latency from minutes to under 1 second while eliminating YAML syntax errors.
- Implemented cross-account AWS provisioning via STS AssumeRole with external ID validation, ensuring zero long-lived customer credentials are stored.
- Designed 18 Django models with complex lifecycle state machines (20 project statuses, 16 deployment statuses) and built a self-correcting deployment pipeline that automatically retries failed CloudFormation stacks after AI-driven template fixes.

Certifications

- **Generative AI Using IBM Watsonx** – IBM Developer Skills Network (Certificate)
- **AWS Certified Solutions Architect – Associate** – Amazon Web Services Training and Certification (Badge)