

Vamsi Alla

Senior AI Engineer | Forward Deployed Engineering | LLM, RAG & Agentic Systems

Remote, USA | vamsialla.dev@gmail.com | +1 (571) 342-9677 | Work authorization: H-1B
vamsialla.com | linkedin.com/in/vamsi-alla | github.com/vamsialladev

PROFESSIONAL SUMMARY

Senior engineer with 12+ years of building production software and 3+ years designing, evaluating and deploying LLM and agentic AI applications for enterprise customers. Works directly with client engineers, product owners and executives to find the real problem, then owns the solution end to end: architecture, APIs, data and retrieval pipelines, the AI layer, evaluation, deployment and handoff. Builds AI that holds up in production: grounded answers, human review for low-confidence outputs, guardrails and regression evaluation.

SELECTED IMPACT

- **~60% faster document processing and ~55% lower model running cost** by replacing a manual enterprise workflow with a multi-agent AI platform, architected and delivered end to end.
- **90%+ field accuracy** on structured extraction in production by routing low-confidence fields to human review instead of letting the model guess.
- **Interactive-speed semantic search** across a large job taxonomy by tuning vector retrieval and metadata filtering in Aurora PostgreSQL with pgVector.
- **Reusable RAG layer, evaluation harness and guardrail configurations** that later engagements adopted as their starting point instead of rebuilding.

TECHNICAL SKILLS

AI & LLM: Anthropic Claude, OpenAI GPT-4o, AWS Bedrock, Azure OpenAI; LangChain, LangGraph, CrewAI; multi-agent orchestration, tool and function calling, MCP, A2A; structured outputs (JSON Schema, Pydantic)

Retrieval, Evaluation & Safety: RAG, hybrid search (BM25 + embeddings), semantic search; pgVector, FAISS, Pinecone; RAGAS, LangSmith, regression evaluation suites; guardrails, PII redaction, prompt-injection defense

Languages & Backend: Python, Java 8/11, SQL, JavaScript/TypeScript; FastAPI, Flask, Spring Boot, Spring AI, REST, GraphQL, microservices, Kafka; React, Streamlit

Data & Integration: Aurora PostgreSQL, Oracle, DynamoDB, Databricks, ETL and embedding pipelines; OpenAPI, OAuth 2.0, mTLS, EventBridge

Cloud & DevOps: AWS (Lambda, Step Functions, API Gateway, S3, EKS, SageMaker, Bedrock, IAM/KMS), Azure, GCP; Docker, Kubernetes, Terraform, GitHub Actions, Jenkins, CloudWatch, Grafana; pytest, JUnit; AI-assisted development (Claude Code, Cursor, GitHub Copilot)

PROFESSIONAL EXPERIENCE

Senior Software Engineer, Enterprise Multi-Agent AI Platform

Aug 2025 – Present

- Own architecture and design decisions for an end-to-end enterprise AI platform on AWS (backend services, data pipelines, APIs and the agent layer) that automates multi-step document work business teams previously did by hand; cut processing time by ~60% and model running cost by ~55%.
- Work directly with client stakeholders, from engineers to executives, to break down workflows, agree on what "good" looks like, and turn it into a technical design and delivery plan.
- Built specialized agents coordinated by a supervisor over A2A messaging and a shared MCP layer, so workflows no single model handled reliably now run start to finish without a human in the loop.
- Built RAG pipelines over FAISS and pgVector and the serverless API layer (Lambda, Step Functions, API Gateway) with automated deployments, IAM/KMS security and observability the client's own team can operate.
- Added guardrails for input and output safety, PII protection and prompt-injection defense, backed by automated evaluation (RAGAS) and regression suites that keep quality consistent across releases.
- Reduced inference cost through a model-agnostic interface (Bedrock, OpenAI, Anthropic), quantization, prompt caching and right-sized endpoints; fine-tuned open-source models where they beat API options on cost.
- Contributed the RAG layer, evaluation harness and guardrail configurations as reusable building blocks, and set team standards for RAG patterns, structured outputs and observability.

Stack: Python, FastAPI, AWS Bedrock (Agents, Knowledge Bases, Guardrails), SageMaker, Lambda, Step Functions, EKS, Aurora PostgreSQL; LangChain, LangGraph, CrewAI; GPT-4o, Claude; pgVector, FAISS; Terraform

Senior AI Engineer, Document Workflow Orchestration

Sep 2024 – Aug 2025

- Built a pipeline that turns unstructured client documents (briefs, emails, transcripts) into schema-aligned JSON, routing low-confidence fields to a human reviewer instead of guessing; held 90%+ field accuracy in production.
- Implemented structured extraction with GPT-4o and Pydantic validation (required-field checks, time-format normalization, confidence flags), and reduced fabricated fields by enforcing explicit "unknown / needs confirmation" behavior.
- Built FastAPI services for client-side integration with client engineers, covering contracts, error handling and throughput.

Stack: Python, FastAPI, OpenAI GPT-4o, Azure OpenAI, LangChain, Pydantic; AWS Lambda, EKS; Docker, Kubernetes

AI Engineer, Job Semantic Search Platform

Feb 2024 – Aug 2024

- Built the embeddings ingestion pipeline and semantic search APIs over Aurora PostgreSQL with pgVector, combining vector similarity with metadata filters and tuning relevance and query performance to interactive response times.
- Added monitoring for ingestion failures, retrieval latency and relevance debugging, and set team standards for RAG patterns, structured outputs, evaluation and observability.

Stack: Python, Azure OpenAI embeddings, Aurora PostgreSQL + pgVector, AWS EKS, S3; REST APIs, Kubernetes, CloudWatch

AI Engineer, Travel Planning Copilot & Virtual Concierge

Apr 2023 – Jan 2024

- Delivered a customer-facing travel assistant with deterministic itinerary generation using structured templates and schema-driven LLM output, integrated with the Google Places API.
- Built and deployed the React chat front end over the RAG pipeline and FastAPI backend, including loading, error and empty states, so the client could use it directly.
- Shipped confidence-based clarification, safe refusal and escalation paths so the assistant asks instead of giving confident wrong answers; hardened the APIs with retries, timeouts and graceful degradation, with latency and fallback metrics and regression prompt suites.

Stack: Python, FastAPI, React, TypeScript; OpenAI GPT, Azure OpenAI, LangChain, Pinecone, pgVector, Pydantic; AWS Lambda, EKS; CloudWatch, LangSmith

Java Developer, Enterprise Microservices Platform · TIAA

Jul 2022 – Mar 2023

- Built Spring Boot REST APIs and microservices, owned deliverables end to end with zero production incidents, and drove SonarQube adoption across the team.

Java Developer, Financial Change Management · Wells Fargo

Feb 2021 – May 2022

- Delivered Spring Boot REST services and microservices as an individual contributor working directly with US counterparts, resolving upstream and downstream issues within SLA.

Java Developer, Product Data Management · Client: Apple

Feb 2015 – Jan 2021

- Built Spring MVC and Spring Boot applications across the full SDLC, with Java/J2EE services over REST and SOAP and a Hibernate data-access layer.

Java Developer, SOA Integration & Healthcare Systems · Napier

Feb 2014 – Jan 2015

- Delivered SOA integrations and REST services for Tier-1 telecom and healthcare clients, including the AAASC authentication and authorization API for Telstra and core modules of the Napier Hospital Information System.

Stack (2014–2023): Java 8, Spring Boot, Spring MVC, Hibernate/JPA, JAX-RS, REST, SOAP, Kafka, Oracle, PostgreSQL, JUnit, Mockito, Maven, Jenkins, SonarQube

PROJECTS

Hotel Operations Knowledge Assistant (independent project)

2026

Live app: hotel-ops-assistant.streamlit.app | Case study and demo video | github.com/vamsialladev/hotel-ops-assistant

- Built a RAG assistant (simulated client engagement) that gives hotel front-desk staff cited, property-specific policy answers, refuses when documents don't say and flags when to check with a manager.
- Measured on a 54-question evaluation (3 runs): 98% answer accuracy on the live deployment and 97% with hybrid search, at \$0.19 per 1,000 questions after a model comparison that cut cost about 5x at equal quality. Visitor testing raised first-question success on a 49-question set from 43% to 100%.
- 274 unit tests (97% coverage), CI with lint, dependency audit and evaluation gates, prompt-injection and XSS defenses, cost caps and owner monitoring; business case with go/no-go pilot KPIs.

Stack: Python, Claude (Haiku), BM25 + embeddings, Streamlit, pytest, GitHub Actions, Docker

EDUCATION

Master of Computer Applications (MCA) and Bachelor of Science (B.Sc.), Acharya Nagarjuna University, India