SAI VIVEKANAND REDDY VANGALA

857-465-9966 | saivivekred@gmail.com | linkedin.com/in/vangalas | github.com/sai-vivekanand

Education

Northeastern University — **Boston, USA**
Master of Science, Computer Software Engineering — Sept 2023 – Dec 2025

BITS Pilani — **Pilani, India**
Bachelor of Engineering, Electronics Engineering — Aug 2016 – June 2020

Technical Skills

Languages: Python, Go, Java, C++, SQL, TypeScript, JavaScript
Backend & Systems: Spring Boot, Node.js, FastAPI, Kafka, REST APIs, TCP, Concurrency, OpenSearch, Airflow
AI/ML Systems: PyTorch, DDP, AMP, Knowledge Distillation, Ollama, LangGraph, RAG, ChromaDB, Gemini Vision
Cloud & DevOps: GCP, AWS, Azure, Kubernetes, Docker, Terraform, GitHub Actions, Linux, CI/CD

Experience

Humanitarians AI — **Boston, USA**
AI Engineer Co-op (Backend & Infrastructure) — Jan 2025 – May 2025

- Achieved **95% multi-modal extraction accuracy** across 10+ document formats by building a **Python RAG** pipeline with Gemini Vision OCR parsing, **LangGraph** orchestration, and **ChromaDB** semantic retrieval.
- Designed a **multi-agent** collaboration framework backed by 4-tier persistent memory using **ChromaDB** and **OpenSearch**, enabling agents to share context and coordinate across distributed document-processing tasks.
- Maintained **99.9% API availability** by automating SLO monitoring and rolling deployments on **GCP** using **GitHub Actions**, **Terraform**, and **Kubernetes**.

Puddl — **Bangalore, India**
Founding Engineer (Volunteer) — June 2022 – Aug 2023

- Architected a zero-trust client-side processing model for an **LLM observability platform**, eliminating server-side API-key exposure by processing OpenAI credentials strictly in-browser.
- Built a **React/TypeScript** analytics SPA backed by **Node.js** telemetry pipelines, ingesting LLM request logs into real-time cost-attribution, latency, and token-usage dashboards.

Blue Yonder — **Hyderabad, India**
Software Engineer — July 2020 – May 2022

- Reduced forecast workbench initial load times by **40%** for enterprise retailers by replacing greedy data fetching with a dynamic **SQL** query builder in the **Java Spring MVC** backend.
- Increased forecasting workflow throughput by **25%** by engineering a server-side **HTTP session caching** layer that enabled planners to run rapid "what-if" simulations without repeated database writes.
- Resolved **70+ customer-reported bugs** annually and achieved **95%+ code coverage** by debugging full-stack Java workflows and building regression test suites with **JUnit**.

Projects

Kafka From Scratch | Go, TCP, Kafka Protocol, Concurrency [GitHub]

- Built a **Kafka-compatible broker** from scratch in **Go**, implementing length-prefixed binary request parsing, correlation IDs, compact encodings, and concurrent client handling over a goroutine-based **TCP** server.
- Implemented core **Kafka protocol** flows for metadata discovery and message reads/writes, parsing **KRaft** cluster metadata and validating behavior through the **CodeCrafters** protocol-compliance test harness.

Distributed Training Systems | Python, PyTorch DDP, AMP, CUDA, Multi-GPU [GitHub]

- Accelerated CNN training by **3.7x** (**92% scaling efficiency**) and scaled throughput to **19,140 samples/second** by implementing **Distributed Data Parallel (DDP)** across a single-node 4-GPU HPC setup with NCCL.
- Reduced GPU training time by **66%** and memory footprint by **31%** while preserving **93% model accuracy** by integrating **Automatic Mixed Precision (AMP)** with bfloat16 autocast for compute-memory efficiency.

Text-to-SQL Distillation Engine | Python, Ollama, DeepSeek-V3, Qwen, GGUF [GitHub]

- Improved schema-grounded Text-to-SQL execution accuracy from **36% to 74%** on a held-out query set by **distilling DeepSeek-V3** outputs into a compact **Qwen 0.6B** student model.
- Reduced serving latency and cost by deploying the quantized student model through **Ollama GGUF**, enabling local SQL generation over a **Pandas/SQLite** schema layer without remote teacher inference.

Auto-Scaling Cloud Web Service | GCP, Terraform, Spring Boot, Pub/Sub [GitHub]

- Architected a 3-tier **GCP** web service with managed instance group auto-scaling, load balancing, health checks, and auto-healing, provisioned end-to-end using **Terraform** and **Packer**-built VM images.
- Decoupled email verification from synchronous API flows using **Cloud Pub/Sub** and Python **Cloud Functions**, reducing request blocking while storing user data in KMS-encrypted **Cloud SQL**.