Akhil S Nair

Portfolio | anair56@uic.edu | 312-804-7555 | LinkedIn | GitHub

Education

University of Illinois Chicago (UIC) ***GPA: 4.0/4.0***
MS in Computer Science *Aug. 2024 – May 2026*
Courses: Computer Algorithms, Machine Learning, Cloud Computing, Android Development, Network-OS programming

APJ Abdul Kalam Technological University ***GPA: 7.57/10.0***
BTech in Computer Science *Aug. 2017 – June 2021*
Courses: Data Structures, Operating Systems, DBMS, Software Engineering, Compiler Design, Computer Networks

Skills

Programming Languages: Java, Python, Go, JavaScript, Groovy, C/C++, C#, SQL, Scala, HTML/CSS
Frameworks & Libraries: React.js, Node.js, Express, Spring Boot, Material-UI, Tailwind CSS, Axios, Apache Spark, Kafka, TensorFlow
Cloud & DevOps: AWS, GCP, Docker, Kubernetes, Git, GitHub, GitLab, Artifactory, CI/CD
Databases: MySQL, Oracle, MongoDB, Postgres, Redis, Elasticsearch
APIs & Protocols: REST, GraphQL, OpenAI Whisper API, Xenova
Developer Tools: VS Code, PyCharm, IntelliJ, Linux, SQS, SDLC, Data Structures & Algorithms

Experience

Amazon Web Services May 2025 – Aug 2025
Software Development Engineer Intern *Seattle, WA*

- Contributed to the **Aurora Serverless v2 – Placement Service** team, designing and implementing migration from **DynamoDB**-based configuration management to **CreaMPuff (CMP)**, an **AWS AppConfig**-based RDS solution, reducing manual per-AZ configuration updates by **90%**.
- Led the integration of **CreaMPuffExtraTastyJavaClient** into the **Java** codebase, creating a new **Guice** module, enabling AZ-aware configuration hierarchy, and executing a safe two-phase (shadow-read → CMP-only) migration, achieving **100% configuration accuracy** in testing.
- Built mismatch detection, deduplication, and cooldown logic for config reads; emitted **CloudWatch metrics** for accuracy tracking and resolved race conditions, cutting duplicate log volume by **80%**, while ensuring reliability with unit tests using **JUnit** and **Mockito**.
- Delivered a full CMP migration, removing **DynamoDB** dependencies, improving deployment consistency across regions, and enhancing operational efficiency through **source-controlled, pipeline-driven** updates.

RxLogix Corporation Nov 2021 – May 2024
Associate Software Engineer II *India*

- Migrated PVCM backend from GSP to **RESTful services**, improving maintainability and reducing load time by **20%**, while maintaining build and deployment pipelines using **Jenkins**.
- Resolved **30+ client-reported issues** on the Continuous Engineering team, ensuring system stability and uptime.
- Managed **WHO and KOREA MFDS Dictionary** integration, optimizing data consistency across **Redis**, in-memory storage, and **Elasticsearch**, improving search efficiency by **15%**.
- Implemented an email intake feature with **Spring AOP**, automating case creation and sending acknowledgments.

Projects

Hybrid Distributed LLM Architecture ↗ | *Scala, Hadoop, Apache Spark, gRPC, AWS EMR, Amazon Bedrock, Docker*

- Processed **100M+ tokens** on WikiText-2 with Hadoop MapReduce for token embeddings and semantic similarity.
- Cut training time by **35%** using Apache Spark + ND4J on AWS EMR; tuned hyperparameters with statistical methods.
- Built hybrid conversational system integrating **AWS Bedrock + Ollama** via gRPC and API Gateway, reducing deployment time by **50%** with Docker pipelines.

TrackerGQL ↗ | *React.js, Passport.js, Apollo Client, Tailwind CSS, Node.js, Express, GraphQL, MySQL*

- Built a full-stack smart expense management app with **React.js**, **Tailwind CSS**, **Node.js**, **Express**, and **MySQL**, featuring expense CRUD operations and interactive charts for spending insights.
- Implemented secure **session-based authentication** with **Passport.js** and optimized API interactions using **GraphQL + Apollo Client**, ensuring efficient and scalable data management.

RPC-Based Transactional Database Engine ↗ | *Go, RPC, Ethos VM, Concurrency Control*

- Engineered a transactional database system in **Go** on **Ethos OS VM**, implementing **Begin/End transaction**, **Get/Put RPCs**, and a custom RPC framework to support multi-client operations.
- Implemented **read/write locks with promotion rules** to ensure sequential consistency, and designed a **logging + recovery subsystem** on non-volatile storage for durability and crash recovery.