Nithish Suresh Babu

(206) 226-8935 | nithish952001@gmail.com | linkedin-nithish-suresh-babu | github.com/nithish-95 | Portfolio

Education

University of Michigan *August 2023 – May 2025*
Master of Science in Computer and Information Science *GPA: 3.78/4.0*

Anna University *August 2018 – May 2022*
Bachelor of Technology in Computer Science and Engineering *GPA 3.7/4.0*

Skills & Interests

Languages & Database: Golang, Python, Modern C++, C, HTML5, JavaScript, MySQL, SQLite3, DynamoDB

Cloud Services & CI/CD: Google Cloud (Cloud Run, Bigtable, Pub/Sub), AWS (EC2, S3, ELB, IAM, App Runner, Route 53, Lambda, Lex, SQS, API Gateway, SNS, DynamoDB), Git, GitHub Workflows, Docker, Kubernetes

AI/ML: Ollama, LangChain, OpenAI, PyTorch, OpenCV

Experience

Research Assistant - Deep Fake (GenAI) *Sep 2024 – Present*
- **AI Detection System Development:** Worked extensively with the GenImage dataset, processing over **1,200,000** real/fake image pairs across **1,000** classes mirroring ImageNet.
- **State-of-the-Art Performance:** Achieved a detection accuracy of **98.5%** in identifying deepfake images, significantly outperforming existing methods by **3.7%** on benchmark tests.
- **Advanced Generator Analysis:** Tested against leading AI/ML models including Midjourney, Stable Diffusion, ADM, GLIDE, Wukong, VQDM, and Big, demonstrating superior detection capabilities across diverse image classes.

Projects

Real-Time Chat Application | *GitHub* *Personal Project, 2024*
- **Technologies Used:** *Go, WebSockets, HTML5, Tailwind CSS, Docker, Amazon DynamoDB, SQS, App-runner, Route53, ACM*
- Engineered a scalable and fault-tolerant real-time chat application inspired by *Discord* supporting **2,500+** users to create or join chat rooms and communicate seamlessly low to **100ms** message latency using Go and AWS.
- Implemented persistent chat history storage by leveraging Amazon DynamoDB with **50ms** read/write latency.
- Designed a **consistent hashing-based routing service** to distribute Web-socket connections across **8+** servers or multiple back-end servers achieving **95%** efficiency.
- Integrated AWS SQS FIFO queues with each back-end server to reliably fan out ordered message delivery among chat room members ensuring ordered delivery for **98%** of users even during **5x** traffic spikes, alongside maintaining persistent chat history across **2** servers with DynamoDB.
- This architecture supports horizontal scaling to accommodate growing user and chat room demands while seamlessly handling server failures. Containerized the system with Docker, reducing deployment times by **40%** through CI/CD pipeline automation.

Tweets Sentiment Analysis using Gen AI | *GitHub* *Final Year Project, 2022*
- **Technology Used:** *Python, HTML5, Tailwind CSS, JS, LangChain, LLama*
- Developed a web application to classify tweets under a specified hashtag and time range into positive, neutral, or negative sentiment categories.
- Implemented real-time sentiment analyzer processing **10,000+** tweets/hour with **92%** accuracy using Llama-2 LLM.
- The application features an interactive dashboard displaying tweet counts for each sentiment and the **top 15** keywords used.
- Utilized the LangChain framework to facilitate smooth integration and interaction with the LLama.
- Optimized LangChain pipelines by reducing per tweet latency by **35%** through prompt engineering and achieving **96.9%** uptime during final demo week with **1.2k+** analyzed tweets.
- The app is deployed as a containerized solution using Docker, ensuring scalability and ease of deployment.

Smart Door with Face Auth | *GitHub* *Personal Project, 2024*
- **Technologies Used:** *Python, AWS (Kinesis Video Streams, Rekognition, Lambda, DynamoDB, S3, SNS)*
- Designed and developed a distributed Smart Door system enabling secure, face-based authentication via video capture using Kinesis Video Streams and achieving real-time face recognition with **98%** accuracy in less than **2s** using AWS Rekognition.
- Owners can register known visitors details. Visitors whose face matches will receive SMS-based **4-digit** OTP's to unlock the door, while details of unregistered visitors are logged in DynamoDB, triggering notifications to the owner for unauthorized access.
- Achieving **99.99%** SMS delivery rate via AWS SNS integration. Ensured robust security by generating unique, single-use OTP's with a 5-minute validity period.
- Built web applications for visitor management, incorporating API's to streamline data capture and authentication workflows.
- Designed visitor management portal to handling **500+** registrations with DynamoDB CRUD operations and **99.95%** API availability using Lambda auto-scaling for peak loads.

Weather Forecast App | *GitHub* *Personal Project, 2024*
- **Technologies Used:** *Go, HTML5, Tailwind CSS, Docker, AWS App runner, Route53*
- Developed a weather application that displays current weather and a **7-day** forecast of the user's current location with **95%** accuracy using browser's location API. Users can also enter the city name, zip code, or coordinates to get the weather of a particular location.
- If permission is not provided, the app will fall back to the user's IP address for the location in real-time.
- Optimized **OpenWeatherMap** integration handling **50+** req/sec with Go concurrent programming. Reduced cold-start latency by **60%** through Docker image optimization on App Runner Achieving **99.9%** uptime with Route53 failover routing and health checks.