Amogh Jagadish Tambad

tambadamogh@gmail.com | +1(480) 876-5096 | *linkedin.com/in/ajtambad/* | *github.com/Ajtambad*

EDUCATION

Master of Science, Computer Science — May 2025
Arizona State University, Tempe, Arizona — GPA: 3.96/4
Relevant Coursework: Cloud Computing, Data Processing at Scale, Data Mining, Data Visualization

Bachelor of Technology (B.Tech), Computer Science and Engineering — May 2021
REVA University, Bangalore, India — GPA: 8.93/10
Relevant Coursework: Data Structures and Algorithms, Computer Architecture, Operating Systems.

SKILLS

- **Languages:** Python, C++, Bash, C, SQL, Scala, HTML, Java, JavaScript, Groovy.
- **Tools and Technologies:** AWS (EC2, ECR, SQS, S3, Lambda, SNS), Git, Jenkins, Kafka, Spark, Heroku, Azure, Splunk, Zabbix, Docker, Kubernetes, PostgreSQL, MongoDB, GitHub Actions, Cribl, OpenShift, Minikube.
- **Libraries and Frameworks:** PyTorch, TensorFlow, Flask, OpenCV, Pandas, Keras, scikit-learn, Nginx, React, Node,js.

EXPERIENCE

IT-Infrastructure-Platform/SRE Intern — Jun 2024 - Aug 2024
Arch Mortgage Insurance, Greensboro, North Carolina

- Filtered logs and events going from OpenShift Kubernetes Clusters to **Splunk** using **Cribl** stream pipelines, reducing Splunk storage utilization by **40-50 GB/day** with **20%** increase in search time.
- Improved readability of Splunk logs with Cribl's Parser and Mask functions, resulting in a concise, easily searchable '_raw' field, reducing parsing time to 2-3 seconds per log.
- Designed a **groovy** script to eliminate **Jenkins GUI** access from the command line, preventing unauthorized access.

System Engineer - 1 — May 2021 - Jul 2023
Oracle Cerner, Bengaluru, India

- Engaged with the software development team on **Splunk** upgrades, troubleshooting, and deployments, ensuring up-to-date servers.
- Migrated 80% data from On-prem to **AWS**, making access to data more flexible, secure, and inexpensive.
- Integrated **Jenkins** and **GitHub** to maintain important documentation and test merge requests for semantic errors, reducing 1-2 hours per week of manual labour.
- Managed **CI/CD** pipelines to automate and oversee 300+ bi-weekly microservice deployments for web applications, enabling rapid delivery of new UI and backend features.
- Managed over 10 projects and 400+ tasks to completion through **JIRA**, resulting in smooth and error-free delivery.

PROJECTS

RAG Implementation for arXiv Papers — Oct 2024 - Nov 2024
Arizona State University, Tempe, Arizona

- Extracted tables, images, equations, and text from **2000+** arXiv papers for vectorization and storage.
- Vectorized and stored them in separate vector stores using models like **CLIP** and text embedding models.
- Implemented **similarity search** to retrieve the top 'k' relevant text and image chunks from **DynamoDB**.
- Summarized retrieved content using the **OpenAI GPT-4o mini** model, delivering concise, contextually relevant responses to user queries.

AWS Based Live Face Recognition App — Feb 2024 - May 2024
Arizona State University, Tempe, Arizona

- Built a web application using **Flask API** and **Gunicorn** server that takes image files as input, performs image recognition, and outputs predictions.
- Created web tier using AWS **EC2** instance that receives images via **HTTP POST** requests and forwards them to AWS **SQS**.
- Designed an **auto-scaling** app tier that spawns up to **20** EC2 instances based on the number of requests in SQS with each of the instances performing image recognition.
- Stored the predictions in **S3** buckets and sent them back through the web-tier, keeping the overall latency at under **3 minutes** for **50** concurrent requests.

Kubernetes based Data Processing Pipeline — Oct 2024 - Nov 2024
Arizona State University, Tempe, Arizona

- Built a highly scalable and available data processing pipeline that allows near-real-time processing and **analytics** of NYC Taxi Rides based spatial document stream data.
- Managed **Kubernetes** deployments of **Kafka**, **Zookeeper**, **Kafka-Connect**, and **Neo4j** components.
- Tested the pipeline using a document stream as input and performed **PageRank** and **BFS** algorithms on the resulting neo4j graph database, establishing individual and relative importance of locations.