Ankush Makhijani

ankushgul.makhijani@sjsu.edu | **+1 856-263-1347** | https://www.linkedin.com/in/ankush-makhijani/ | https://github.com/wolfy5786

Work Experience

AceMyTravel | Remote June 2025 – October 2025

Software Engineer in AI |Python, FastAPI, Docker, React, GCP NLP, Jenkins, Lang chain

- Working in a fast-paced startup, I focused on designing and prototyping a system that uses LangChain prompt flows and lightweight agents to turn user prompts and live travel data into personalized tour itineraries.
- Implemented **gRPC** based services for booking updates and inter service communication which reduced API call time to less than **70%** on avg.
- Applied **Google Cloud NLP** API for text analysis and semantic search, boosting search relevance by **28%**. Deployed and managed services on **GCP** with **Docker**.
- Collaborated with cross-functional teams to design, prototype, and deliver customer-facing features. Optimized performance with caching, async processing, and reactive design, reducing average response time by **19%**.

Rutgers New Brunswick | New Brunswick, NJ May 2024 – October 2024

Research Assistantship |Linear Programming, Data visualization

- Implementing polynomial approximation for Log1p for all rounding methods for 32-bit floating point, which was not only more accurate on **7 million inputs** but **1.4 times faster** than the current default library **'libm'.**
- Randomly select a few generated constraints and train polynomial using **Linear programming,** we test the polynomial on another set of randomly selected constraints and retrain our polynomial on failed test cases.

Simplified IT Solutions | Thane, India April 2023 –August 2023

Software Intern | JAVA, MYSQL, REST API, Spring Boot, Swagger

- My key role was in the development and maintenance of middle-tier services using **Java with Spring Boot** and **REST APIs** to support a scalable web application architecture.
- Leveraged Test Driven Development to design thorough **JUnit/Mockito** tests, automated the pipeline, and contributed to faster development cycles and more dependable releases.
- Validated data contracts between APIs and frontend components, detecting several critical bugs early in the development cycle, preventing production issues.

Education

San Jose State University | US May 2027

Masters of Science | **Major: Software Engineering | Sp: Enterprise Software & Distributed Systems**

Rutgers University, College of Arts and Science | US May 2025

Bachelor of Science | **Major: Computer Science**

Granted with the Chancellor's Merit Scholarship at Rutgers University. **Dean's List**

Skills

Mathematics & Data Science: Probability, Graphs, Statistics, Linear Algebra, Calculus, Machine Learning Algorithms.

Programming Languages: Python, Java, C++, JavaScript, SQL, TypeScript, GO

Cloud: AWS (Lambda, DynamoDB, S3, EC2, RDS, API Gateway, CloudWatch), GCP(compute Engine, cloud SQL, NLP)

Database: MySQL, PostgreSQL, MongoDB Neo4j

Backend: Spring Boot, Node.js, Hibernate, Flask, Fast API, JDBC, Django, Redis, Kubernetes

Messaging: ZeroMQ, RabbitMQ, Kafka

CI /CD: GitLab, Github Actions, Jenkins, Docker

Software Testing: JUnit, Mockito, PyUnit, Selenium, Playwright

Technical Skills: Algorithm and logic development, Agile Methodology, Git Version Control, Operating Systems and System Design.

Projects

Piggyback Taxi booking REST service | JAVA, PostgreSQL, Spring boot, Docker, AWS EC2 April 25 – July 25

- Built a Backend cab booking REST service with **Spring Boot REST APIs** and **PostgreSQL** Database.
- Designed **Microservices Architecture** for each service like user authentication, user login/sign up, cab booking, ride tracking, and payments service, deploying each server on an Amazon **EC2** instance with an additional **API gateway.**
- Ensured API security, error handling, and scalability using **MVC architecture**, documented APIs with **Swagger**, **POSTMAN** for testing.

Online Multiplayer Poker | Python, Django, Redis, MongoDB, Docker August 25 – August 25

- Developed a real-time multiplayer poker game using **Django Channels** and **WebSockets** with **Redis** for scalable socket management, and built a responsive **React + Tailwind UI** with live updates and seamless player interaction.
- Integrated Multiplayer lobby, with in game chatting, game replays and logs.
- Containerized frontend, backend, and Redis services using **Docker** and orchestrated them with docker-compose.