Chandrakant Koneti

984-374-9958 | chandrakantkonetius@gmail.com | LinkedIn | GitHub | Raleigh, NC, USA

Education

North Carolina State University Aug 2024 - **May 2026**
Master of Science, Computer Science *Raleigh, NC, USA*
Coursework: Data Structures & Algorithms, Object Oriented Design, Operating Systems, Computer Networks, AI, ML

Indian Institute of Engineering Science and Technology Nov 2020 - May 2024
Bachelor of Technology, Information Technology *Kolkata, India*

Technical Skills

Languages: Java, Python, JavaScript, TypeScript, C++, HTML, CSS, SQL
Frameworks: Spring Boot, React, FastAPI, Node.js, Redux, Material-UI, Webpack, Temporal
AWS Cloud: Lambda, API Gateway, IAM, DynamoDB, S3, Cognito, Bedrock, CloudWatch
Databases: PostgreSQL, MongoDB, Redis, MySQL; **Messaging**: Apache Kafka, RabbitMQ
Dev Tools: Git, Docker, Kubernetes, CI/CD, Postman, Swagger, Linux, Agile (Kanban, Scrum), Jira, Claude, Cursor
Achievements: Top 2% Coder (Knight Badge) on LeetCode, Meta Front-End Developer – Coursera *Certificate*

Experience

Software Engineer Intern May 2025 – Aug 2025
Reva.ai *Remote, United States*

- Built and deployed 12 **Python Lambdas** using API Gateway, leveraging **AWS** DynamoDB as global storage for multi-tenant config management, eliminating 30GB data across **20+** Kubernetes clusters and saving $2K/year.
- Increased reliability of long-running **AI workflows** by enhancing the core **Java** Spring Boot Temporal microservice, unifying error handling for **59** activities and reducing manual retries by 98%.
- Reduced stream API latency by 120 ms by multithreading customer usage logging in a **FastAPI LLM** service that calls Amazon Bedrock via LangChain, streaming events to **CloudWatch** for real-time billing.
- Enabled real-time sync of 100% **SSO** users (Okta, Microsoft, Google) to on-premises databases via an Lambda triggered by AWS **Cognito** post-signup events, invoking a custom REST API.
- Boosted reliability of integration testing by 40% with real mock **events** for Lambdas and Spring Boot services.

Frontend Engineer Intern Sep 2023 – Dec 2023
RabbitLoader *Remote, India*

- Led the migration of core micro-frontend from **React** v14 to v18, fixing legacy bugs, preventing regressions in prod.
- Cut load time from 12s to 3.5s and bundle size from 300MB to 3MB by re-architecting with **Redux** state and optimizing load distribution in **Webpack** to efficiently route modules across Angular and Vue apps.
- Boosted UI performance by 400% with infinite scroll and lazy loading, delivering components **5×** faster using **MUI**.
- Developed mock APIs in **Node.js** for **E2E** testing, reducing **QA** cycles by 30% through automated **CI/CD** validation.

Software Developer Intern May 2023 – Jul 2023
Cambridge Technology *Hyderabad, India*

- Built 2 Java Spring Boot **microservices** to ingest, validate, and serve **2M+**/yr medical-diagnosis records.
- Enabled reliable real-time medical data streaming through a **Spring** service integrated with **RabbitMQ** for user notifications and **Kafka** for ML & ETL pipelines, preventing over 10GB of data loss during peak loads.
- Reduced **Java** GET API latency by 2s and reduced database calls by 65% through **Redis** caching of aggregated 100K+/week records, sourced from PostgreSQL and MongoDB.
- Boosted code coverage by **46%** and cut prod bugs by implementing **unit** and integration tests with JUnit, **Mockito**.

Projects

AI Low-Code UI Generator Tool - Weberator | **IEEE Publication** - *RESEARCH PAPER* June 2024

- Enabled **Figma**-style drag-and-drop, translating designs into an N-ary UI tree mirroring layout hierarchy; applied prompt engineering on **OpenAI API** to generate production-ready **React** code, cutting dev time by 55–65%.
- Implementing lossless, **batched** payloads for durable DB writes, boosting throughput and preventing **data loss**.

ML based Drawing Application - DrawIt | *Code Repo*

- Building a multiplayer drawing game with React, Redux, and **Material-UI**, integrating **ML5.js** for real-time canvas image processing and implementing custom logic for accurate scoring.
- Optimizing real-time gameplay using **Express.js**, **Socket.io**, caching, and secure sessions with **JWT**.