DHANASEKAR RAVI JAYANTHI

+1(669) 319-9749 | dhanasekar78788@gmail.com | LinkedIn | Santa Clara,CA

Summary

Software Engineer building **modern web applications** with **React + TypeScript**, **REST APIs**, and **SQL**. Strong in **frontend engineering** (component design, state, accessibility, performance) with solid backend fundamentals (**Node/Java/Python**) and database work (**PostgreSQL**). Hands-on with **Git/GitHub**, **Docker**, and **CI/CD (GitHub Actions)**, plus cloud-based deployments and troubleshooting with logging/metrics.

Education

Master of Science in Computer Science and Engineering, **Santa Clara University,** USA — **Sep 2023 - Jun 2025**

Bachelor of Engineering in Computer Science and Engineering, **Anna University,** INDIA — **Aug 2019 - Jun 2023**

Relevant Coursework: Data Structures & Algorithms, Distributed Systems, Advanced Operating Systems, Database Management Systems, Computer Networks, Artificial Intelligence, Machine Learning.

Skills

Languages: Java, Python, TypeScript/JavaScript, SQL

Frontend: React, HTML5, CSS3, REST API Integration, Accessibility (a11y), Performance Optimization

Tools/DevOps/Testing: Git/GitHub, Docker, CI/CD (GitHub Actions), Postman, Jest/JUnit, Linux/Unix, AWS (Lambda/S3 basics)

Professional Experience

Software Engineer - TipTop Technologies, Inc. — Oct 2025 - Present

- Built and shipped **React + TypeScript** UI features, improving user workflows and reducing UI defects through iterative releases.
- Integrated **REST APIs** and optimized client/server performance using caching and request tuning, improving responsiveness and reliability under real user traffic.
- Added **observability** (structured logs/metrics/health checks) and strengthened quality with automated tests and CI checks, improving debugging speed and release confidence.

Software Developer - Frugal Innovation Hub — Jan 2025 - Jun 2025

- Built backend services and **automation workflows** with **REST APIs** and **PostgreSQL**, supporting production pipelines and validations.
- Optimized **SQL queries and indexes**, reducing latency by **~30%** and improving throughput for analytics and core workflows.
- Built **CI/CD pipelines (GitHub Actions)** with automated tests and checks, improving release consistency and troubleshooting via logs/health endpoints.

Software Engineer - Ecopez Technologies Private Limited — Mar 2023 - Aug 2023

- Built high-volume **Node.js + PostgreSQL** services and REST endpoints, improving throughput by **~35%** using batching, caching, and indexing.
- Built **queue-based async workers** with retries and failure handling, supported by **logging and health checks** for reliability and recovery.
- Improved deployment reliability using **Docker-based workflows** and repeatable builds, reducing deployment errors by **~45%**.

Projects

Context-Aware Email Assistant (RAG)

- Built a retrieval-augmented generation (**RAG**) pipeline using **LangChain + ChromaDB** to enable semantic search and accurate email/document summarization at scale.
- Fine-tuned **Gemma** using **LoRA (Dolly-15K)** and optimized inference with parallel processing + caching, reducing end-to-end response latency by **~30%**.
- Shipped **serverless REST APIs (AWS Lambda)** and a **React monitoring dashboard** to track latency/quality metrics and speed up debugging during runs.

Stock Market Publish–Subscribe Simulator

- Built a distributed **pub-sub system** in **Java** using **multithreading + sockets** to simulate real-time event streaming.
- Implemented **thread-safe** publishers/subscribers with **topic-based routing** and resilience controls under concurrent load.
- Containerized with **Docker** and automated builds/tests using **GitHub Actions** for CI/CD and repeatable releases.

Cloud Data Movement & Validation Pipeline

- Built a Python **ETL/data movement pipeline** to move data from **PostgreSQL → S3**, including schema checks and row-count validation to ensure correctness and reliability.
- Automated scheduled runs using AWS Lambda, added CloudWatch logging/alarms and retries for failure recovery, improving observability.
- Packaged execution with **Docker** and enforced build/test gates using **CI/CD (GitHub Actions)**, improving repeatability and reducing deployment mistakes.