Laksh Jethani

lakshjethani.dev | +1 (352) 256-5058 | ljethani@usc.edu | linkedin.com/in/laksh-jethani | github.com/lakshjethani

Education

University of Southern California — Aug 2024 – May 2026
Master of Science, Computer Science — *Los Angeles, CA*
Relevant: Data Structures and Algorithms, Natural Language Processing, Web Technologies

University of Mumbai — Aug 2018 – May 2022
Bachelor of Engineering, Computer Science — *Mumbai, India*
Relevant: Parallel and Distributed Systems, Network Security, Operating Systems

Experience

Software Engineer Intern, *Supplyframe, a Siemens Company* — Los Angeles, CA — May 2025 – Present

- Rebuilt dev.hackaday.io and Hackaday.io into a modular, **Storybook-driven** system using **React/Angular** and **AJAX**, extending backend workflows in **Node.js/Express**; improved UI consistency and sped up design–QA cycles by **25%**.
- Cut **p95 API latency by 20%** (480ms → 380ms) through exponential-backoff retry logic for Swagger tokens and a tighter Solr payload flow, lowering request volume and boosting responsiveness on high-traffic paths.
- Rolled out a hardened **OAuth2 sign-in flow** with **CSRF safeguards** and **Redis-based scope caching** for dev.hackaday.io **(1M+ users)**, closing replay vectors and strengthening auth consistency.

Senior Software Development Engineer, *Deutsche Bank* — Pune, India — Aug 2023 – Jul 2024

- Engineered a **Kafka-driven retry pipeline** with idempotency keys, transactional outbox, DLQs, and jittered offset-managed retries, lifting payment success by **21%** on a **$300M+/day** platform and lowering reconciliation and on-call escalation rates.
- Migrated Fabric from on-prem to **GCP** through Infrastructure as Code and seamless blue-green deployments, refining autoscaling rules and multi-zone topology to lower run cost by **17%** and strengthen failover resilience across three zones.
- Integrated **Azure MFA** with a custom **OIDC** provider to federate external users, enforce Conditional Access with step-up authentication for sensitive scopes, and unify claims and token lifetimes, leading to fewer login issues.
- Mentored two new grads through code reviews and debugging sessions, accelerating onboarding and raising overall PR quality.

Software Development Engineer, *Deutsche Bank* — Pune, India — Aug 2022 – Jul 2023

- Maintained SLAs on critical paths with **circuit breakers**, idempotent retries, and full **observability** (metrics, logs, tracing).
- Decomposed a monolith into **scalable Spring Boot microservices**, adding thread-safe **concurrency controls** and circuit breakers for resiliency, eliminating race conditions and lowering failure rates.
- Enhanced **React + Redux** architecture with optimized state handling, memoization, and lazy loading; added GraphQL/REST integration with caching and pagination, cutting re-renders by **40%** and boosting responsiveness by **25%**.
- Led UAT cycles with 10+ users and analysts, prioritizing feedback to enable delivery of **3 key features** in the next release.

Software Development Engineering Intern, *Deutsche Bank* — Pune, India — Jun 2021 – Aug 2021

- Built a sharded cache with **Redis Cluster** and Spring Boot, applying consistent hashing, Raft-style leadership, and quorum reads/writes to scale to **millions of keys** and serve **10+ downstream services** without data loss during node failures.
- Applied **Merkle trees**, gossip protocols, and **Bloom filters** to prevent cache stampedes and cut redundant calls.

Selected Projects

Distributed Observability Pipeline — Go, Kafka, OpenTelemetry, Elasticsearch (GitHub) — Feb 2025 – Apr 2025

- Engineered a distributed observability pipeline as a self-initiated grad project in **Go**, with Kafka ingesting and indexing high-volume **OpenTelemetry** traces and logs (millions of events/minute) into **Elasticsearch** for real-time analysis.
- Deployed on **Kubernetes** with fine-grained **RBAC** for scalable and secure operations; built in fault tolerance and automated alerting, resulting in **99.9%** pipeline uptime.

MCP SafeSQL — TypeScript/Node.js, SQLite/Postgres (GitHub) — Dec 2024 – Jan 2025

- **Model Context Protocol server** exposing a read-only `sql.query` with allowlisted views, parameterized queries, and PII masking across **40 + columns**; adds RBAC, audit logs, pooling, and OpenTelemetry for SQLite/Postgres
- **Safe EXPLAIN**: Returns the operator tree with cost and row estimates without executing, redacts identifiers and literals, fingerprints statements for caching, and flags full scans or missing indexes on **50+** test queries

Technical Skills

Languages: **Java**, **C++**, Go, Python, Android (Kotlin), **TypeScript**, JavaScript, SQL, HTML, CSS
Backend: **Spring Boot**, **Node.js**, Express, REST, **Apollo GraphQL**, Django, JPA, Hibernate, OpenAPI
Frontend: **React**, Redux, EJS, SCSS, Storybook, **AJAX**, Bootstrap, WCAG 2.1
Data & Messaging: **PostgreSQL**, MySQL, MongoDB, **Redis**, Apache Solr, **Kafka**, Celery
Cloud & DevOps: **GCP**, **AWS** (EC2, S3, Elastic Beanstalk), **Docker**, **Kubernetes**, GitHub Actions, Jenkins
Observability & Quality: **OpenTelemetry**, distributed tracing, JUnit, Mockito, Jest, Playwright, ESLint, Prettier, Nginx
Auth & Security: **OAuth2**, **OIDC**, CSRF, Azure MFA