SAGAR SHETTY

Senior Engineer

+91 9867894744 | sagar.shetty381@gmail.com | LinkedIn | GitHub | Mumbai, India | Portfolio

PROFESSIONAL SUMMARY

6+ years Senior Full Stack Engineer skilled in React, Next.js, Node.js, and TypeScript. Designs microservices-based architecture, scalable APIs, improves performance with Redis caching, and builds resilient systems on AWS. Mentors teams, elevates Core Web Vitals, and delivers maintainable code with focus on clean architecture, API design, and data modeling.

TECHNICAL SKILLS

Programming Languages: JavaScript, TypeScript, Python, SQL, GraphQL
Frontend: React.js, Next.js, Tailwind CSS, Redux, AngularJS
Backend: Node.js, Express.js, REST APIs, GraphQL APIs, Microservices Architecture, WebSocket, Kafka
Tools: Jenkins, Git
Databases: PostgreSQL, MySQL, MongoDB, Redis
Platforms: AWS, EC2, S3, CloudSearch
Others: DSA, System Design, API Optimization, Problem Solving, CI/CD

PROFESSIONAL EXPERIENCE (6+ YEARS)

Senior Engineer | *Aug 2022 - Present*

BTS | *Mumbai*

- Built modular frontend and backend components with React, TypeScript, Node.js, and REST, applying **clean architecture** and **secure patterns**.
- Engineered high-throughput APIs and **caching** via Redis and asynchronous processing, reducing API response times by 70% and accelerating user journeys.
- Implemented **AWS CloudSearch-based** indexing pipeline, reducing search latency by 90% and increasing click-through rate on recommendations by 28%.
- Directed large-scale **data migrations** to cloud storage with zero data loss and minimal downtime, enhancing reliability and scalability.
- Mentored engineers on React patterns, API design, and system design; led **code reviews** and **architecture sessions** to enforce maintainability.
- Maintained **unit and integration tests** to protect feature releases and refactors, improving stability and reducing regressions.
- Collaborated with product and operations for **cross-functional alignment**, contributing to 25% improvement in feature adoption.
- Established code review checkpoints and expanded test suites, boosting unit test coverage from 60% to **92%** and cutting release defects by 40%.

Technologies Used: *React.js, TypeScript, Node.js, REST APIs, Redis, MongoDB, AWS CloudSearch, Git*

Backend Engineer | *Aug 2019 - Aug 2022*

TCS | *Mumbai*

- Delivered trading features and automated market-close settlements with **shell scripts**, cutting manual intervention by 40% and improving processing speed by 30%.
- Orchestrated **release management** and **system testing** for critical deployments, ensuring smooth production rollouts and operational continuity.
- Served as **Shift Lead**, delegating tasks and managing resources to maintain team efficiency and reliable on-call operations.
- Optimized components within the **trading architecture**, achieving 35% improvement in throughput and responsiveness.
- Partnered with business and release stakeholders to align deployments with trading windows, reducing release lead time by 18% and accelerating time-to-market.

Technologies Used: *Shell scripting, SQL, Linux, Git*

PROJECTS

Spark – Real-Time Dating Platform (Personal Project)

https://github.com/sagarshetty381/bechain-web

Full-stack dating platform using React, Node.js, PostgreSQL, Socket.io, Kafka, and WebRTC. Real-time messaging with event-driven architecture focused on scalable APIs and performant UI.

- Architected **event-driven messaging** platform using Kafka, Socket.io and WebRTC; load-tested APIs to handle 450 RPS with p95 latency under 180ms while reducing DB queries per request from 6 → 2
- Implemented **fault-tolerant** event handling with Kafka retries and dead-letter queues, ensuring zero message loss during simulated service failures.

Pocketpedia (Personal Project)

https://github.com/sagarshetty381/gsmarena-graphql

Node.js and GraphQL backend emphasizing schema design and efficient data fetching to minimize over-fetching and improve query performance.

- Introduced **server-side caching** and intelligent query batching to limit redundant web scraping and minimize rate limits, improving API throughput to ~250 QPS under simulated load while reducing origin scraping frequency by 60%.
- Added **pagination, filtering, and error-handling** mechanisms, cutting average API response sizes by 45% and decreasing tail latencies (p95) from ~450ms to ~180ms in benchmark runs.

EDUCATION

B.E. in Information Technology | *Mar 2019*

Terna Engineering College | *Navi Mumbai, India*