Garv Aggarwal

New Delhi, India | +91-92133-10877 | aggarwalgarv0505@gmail.com
LinkedIn | **Github** | **Portfolio**

Backend Software Engineer

Backend engineer with experience building **Java Spring Boot** services on a **54-service production platform**, focused on **distributed systems**, async pipelines, and **production reliability**. Strong in system design, concurrency, and performance profiling.

TECHNICAL SKILLS

Languages: Java, Golang, JavaScript, TypeScript
Backend: Spring Boot, Node.js, Microservices, REST, WebSockets, Event-Driven Architecture
Distributed Systems: Redis, ZooKeeper, RabbitMQ, Kafka, MongoDB
Cloud & DevOps: **AWS (Lambda, EC2, S3, CloudWatch),** Docker, Docker Compose, CI/CD, Linux, Nginx
Concurrency & Systems: Goroutines, Channels, Worker Pools, Thread Safety, Connection Pooling
Tools & Practices: Git, System Design, Performance Optimization, Profiling, Monitoring, Container Orchestration

EXPERIENCE

Octro Inc — Member of Technical Staff (Backend Engineer) | *Mar 2025 – Present*

- Designed and operated **Java Spring Boot** microservices on a **54-service high-traffic production platform** — WebSocket sessions, async event pipelines via RabbitMQ, Redis caching, and MongoDB as primary store.
- Made services configurable via **ZooKeeper runtime config** — feature flags propagated to **54 service instances without redeployment**, enabling zero-downtime staged rollouts and instant kill-switches across the fleet.
- Re-architected a blocking notification call on the auth critical path into a fully async **Amazon SQS pipeline** — per-platform routing, ZooKeeper rate limits, **dead-letter queue, and CloudWatch** alerting.
- Built a **multi-provider OAuth** account linking system across 3 microservices — time-bound token generation, email verification, and **atomic merge** with **idempotency guards** and **partial-failure rollback**.
- Resolved **6 production incidents on-call**: traced a **Redisson distributed-lock thread-affinity violation (IllegalMonitorStateException)** recurring over weeks — fixed root cause, ending repeated service restarts across 3 services; resolved a WebSocket race causing duplicate session auth.
- Eliminated **N+1 patterns**: batched Redis reads on a high-frequency polling service (every 10s per user) and MongoDB fan-out on the login path. — **cutting DB round-trips on the hottest paths**. .
- Built a **polyglot** compute layer bridging Java microservices and **versioned TypeScript AWS Lambda functions** — type-to-version resolution from runtime config, async invocation via **CompletableFuture**.
- Recognized as **Best Performer** for delivering high-impact backend contributions.

StepOut — Software Developer Intern *| Nov 2024 – Feb 2025*

- Built a **Flask video processing pipeline** — ingested match footage, ran per-frame **YOLO object detection** to identify and annotate players with jersey numbers, and re-encoded processed frames into a final output video.
- Implemented **asynchronous background upload/download** for large media files enabling uninterrupted transfers; reduced redundant API calls on data-heavy screens and improved **state synchronization**, cutting unnecessary network round-trips.

EDUCATION

Maharaja Agrasen Institute of Technology, New Delhi — **CGPA: 9.0**
B.Tech — Computer Science & Engineering (Minor: AI/ML) — **2021 - 2025**

PROJECT

Code Execution Engine | Go, Docker API, Goroutines, SSE, Redis Pub/Sub

- Built a LeetCode-style execution engine in Go — **worker-pool architecture** (goroutines + buffered channels) supporting **500+ concurrent submissions**; each runs in an isolated Docker container with resource limits and context-based timeouts, cutting cold-start from **~3.2s to ~450ms**.
- Streamed real-time build output over **Server-Sent Events**, removing polling and cutting client-server bandwidth by **~70%**.

Load Balancer | Golang, sync.RWMutex, fsnotify, YAML

- Built a production-style HTTP load balancer in Go — multiple routing strategies, **concurrency-safe request path, zero-downtime config reloads** via fsnotify, and active health checks with failure isolation.