Magomed Bonkurov

Senior Java Developer

Batumi, Georgia · Remote, open to relocation · +7 938 902-06-52 · antichit95@gmail.com · Telegram: @mnstrvu · bonkurov.ru

SUMMARY

Senior Java developer with 6 years and 5 months of commercial Java experience, almost 4 of them at senior level. I work in EdTech and, more broadly, with user-facing services where the load arrives in sharp peaks. 3 years and 5 months of that were on the Sberclass learning platform — I joined it through a contractor and later moved in-house at Sberobrazovanie.

Main stack: Java 17, Spring Boot, PostgreSQL, Apache Kafka, Redis, Docker, Kubernetes and GitLab CI/CD.

The change I point to most often: I brought assignment delivery down from 1.9 s to 260 ms at p95 by removing an N+1 and adding a composite index. I usually own a service from end to end — data model, integrations, releases, production on-call — and I mentor junior and middle developers and run technical interviews.

KEY SKILLS

Languages and platforms: Java 17 (records, sealed classes, text blocks), Java 8 and Java 11, including a migration from 11 to 17; SQL; Linux.

Backend: Spring Boot, Spring Framework (Core, MVC, Data JPA, Security), Hibernate/JPA, JdbcTemplate, REST APIs, GraphQL (including DataLoader), OpenAPI/Swagger, Keycloak, OAuth 2.0 / OIDC / JWT, Resilience4j.

Data and integrations: PostgreSQL — EXPLAIN (ANALYZE, BUFFERS), pg_stat_statements, composite indexes, N+1 removal, batching, declarative partitioning, materialised views; HikariCP, Flyway; Redis — cache-aside, TTL and event-based invalidation; Apache Kafka — transactional outbox, at-least-once delivery, idempotent consumers.

Infrastructure and testing: Docker, Docker Compose, Kubernetes (I run services in it: deployments, ConfigMaps, Secrets, Helm charts, kubectl; the cluster itself is managed by a platform team); GitLab CI/CD, SonarQube, Maven, Gradle, Git; JUnit 5, Mockito, Testcontainers, JMeter; Grafana, Kibana/ELK, SLF4J and Logback.

Practices: Microservice architecture, idempotency and duplicate handling, caching strategies, working with legacy code, code review, mentoring, Scrum.

EXPERIENCE

Senior Java Developer, Backend — Kub, Grozny (remote)

March 2025 — present (1 year 6 months)

Kub is a software development and system integration company. An internal platform for processing requests and billing for corporate clients: it takes a request in, routes it by business rules, calculates the charges and exchanges data with external systems. I own the backend of three services that carry their peak load during business hours.

- Replaced synchronous REST calls between services with asynchronous messaging over Apache Kafka. On the producer side I used a transactional outbox with a poller; on the consumer side, an idempotent consumer that deduplicates by event key in a `processed_events` table, written in the same transaction as the business operation and kept for one month. Events are no longer lost when a neighbouring service is down. I accepted eventual consistency as a deliberate trade-off and chose it over a distributed transaction because of the coupling and the support cost.
- Migrated an event table that had grown to hundreds of millions of rows to declarative monthly partitioning: I moved the historical data in batches and attached it with `ATTACH PARTITION` without downtime, and automated the creation of new partitions and the removal of old ones. Analytical queries no longer compete with OLTP traffic.
- Wrapped a synchronous external call in Resilience4j (timeout and circuit breaker), so that degradation in a neighbouring service no longer exhausts the thread pool and brings the whole endpoint down.
- Introduced mandatory two-reviewer approval for the core modules, plus a quality gate in GitLab CI/CD: the build fails on critical SonarQube issues and when coverage drops below the threshold, and integration tests run on Testcontainers.
- Mentor junior and middle developers: weekly one-to-one meetings, growth plans and review feedback sessions. I also run technical interviews and wrote the checklist for the Java and SQL section.
- Run services in Kubernetes (deployments, ConfigMaps, Secrets, Helm charts, pod diagnostics with kubectl) and release to dev, staging and production through GitLab CI/CD. I take part in production on-call: after an incident where the connection pool was exhausted at peak time, I added alerts on pool saturation and Kafka consumer lag and wrote the postmortem. The incident has not happened again.

Stack: Java 17, Spring Boot 3, Spring Data JPA, PostgreSQL, Apache Kafka, Redis, Kubernetes, Helm, Docker, GitLab CI/CD, SonarQube, REST API, Resilience4j, Flyway, JUnit 5, Mockito, Testcontainers, Grafana, Kibana, OpenAPI/Swagger, Gradle, SLF4J and Logback.

Senior Java Developer, Backend — Sberobrazovanie (Sberclass product), remote

October 2022 — February 2025 (2 years 5 months)

I moved from the contractor Haiku dev into the client's own product team: the same product, but with wider responsibility. Sberclass is a learning platform for school students, with a sharp traffic peak in the first minutes of every lesson. I owned the assignments and progress module end to end, from gathering requirements with the product manager to release and production support.

- Reduced the latency of the assignment card from 1.9 s to 260 ms at p95. The card was built with more than 300 queries (N+1 on nested entities), so I replaced them with a batch fetch using join fetch and added a composite index on `(student_id, task_id)`. I measured the result in JMeter and confirmed it in Grafana on production during the peak minutes.
- Ran load tests of the assignment delivery endpoints in JMeter. The first bottleneck was the HikariCP pool of 10 connections, and the second was a synchronous call to the content service. I sized the pool from the real concurrency, keeping the database `max_connections` limit in mind, and made the call asynchronous — throughput roughly doubled while p95 stayed within the SLA.
- Cached user profiles and reference data in Redis (cache-aside, with a TTL as a safety net for missed invalidation events and a jittered TTL so that many keys do not expire at the same time during peaks). This removed most of the repeated reads from PostgreSQL during school hours.
- Built a GraphQL layer on top of Spring Boot and removed the N+1 problem in the resolvers by batching through DataLoader — the number of database calls per card dropped from dozens to two or three. I also added query depth and complexity limits, so that a single client query cannot bring the service down.
- Moved the progress aggregates into a materialised view with a nightly `REFRESH CONCURRENTLY` (supported by a unique index): a class report went from about 1.5 minutes to a few seconds, and reporting no longer competes with OLTP.
- Implemented REST APIs for the assignments, progress and test results modules, moved authorisation to Keycloak (OAuth 2.0 / OIDC / JWT) with a role model for students, teachers and administrators, and documented the contracts in OpenAPI. I covered the module with integration tests on Testcontainers (real PostgreSQL and Kafka), ran code reviews and onboarded two developers into the module.

Stack: Java 11 → 17 (I took part in the migration), Spring Boot 2.x, Spring Data JPA, JdbcTemplate, PostgreSQL, Apache Kafka, Redis, Keycloak (OAuth 2.0 / OIDC / JWT), GraphQL, REST API, Docker, Bitbucket, Bitbucket Pipelines, Maven, Gradle, Flyway, JUnit 5, Mockito, Testcontainers, JMeter, Grafana, Kibana, OpenAPI/Swagger.

Java Developer (middle), Backend — Haiku dev, Saint Petersburg (remote)
October 2021 — September 2022 (1 year)

Outsourced development on the Sberclass product: an outstaff team working on the content and assignments module. After a year I joined the client's in-house team, working on the same product with wider responsibility.

- Found and fixed a memory leak. A custom cache built on `ConcurrentHashMap`, with no size limit and no TTL, was holding on to user session objects. I took a heap dump with `HeapDumpOnOutOfMemoryError` and analysed it in Eclipse MAT, where that cache turned out to be the dominator. I replaced it with Caffeine (`maximumSize`, `expireAfterWrite`) and added an alert on old gen usage. The service stopped running out of memory and the scheduled restarts were cancelled.
- Delivered features of the assignments module end to end: REST controllers and GraphQL resolvers, schema migrations, unit and integration tests, releases to dev and staging, and acceptance with the analyst. I also cleaned up flaky tests, so CI runs stopped failing without any code changes.
- Investigated production defects through the logs in Kibana. The hardest one: a double submit by a student created a duplicate result. I fixed it with a unique index on the business key and an idempotency key on the request, so a repeated submit now returns a conflict together with the current state.

Stack: Java 8 and 11, Spring Boot 2.x, Spring Data JPA, JDBC, PostgreSQL, GraphQL, REST API, Caffeine, Eclipse MAT, Maven, Bitbucket, JUnit 5, Mockito, Kibana, SLF4J and Logback.

Java Developer (junior → middle), Backend — ChemCool, Grozny · full-time (remote)
November 2019 — April 2021 (1 year 6 months)

An educational portal for students in grades 8 to 11: tests and knowledge assessment. A small team and a product running in production.

- Built the backend of the client-facing service on Spring Boot: the topic catalogue, test taking and delivery of results. I accessed PostgreSQL through Spring Data JPA and Hibernate and documented the contracts in OpenAPI/Swagger.
- Set up the exchange of events between services over Apache Kafka: test results were sent to the statistics service asynchronously, so recalculation no longer blocked the response to the user.
- Implemented authentication and authorisation with Spring Security: JWT login, student and teacher roles, and a rule that blocks access to other users' results. I covered my modules with unit and integration tests, maintained the module READMEs, and made small changes to the React frontend when debugging my own features end to end.

Stack: Java 8, Spring Boot 2.x, Spring MVC, Spring Data JPA, Spring Security, Hibernate/JPA, PostgreSQL, Apache Kafka, REST API, Maven, Git, JUnit 5, Mockito, OpenAPI/Swagger, SLF4J and Logback; basic frontend work in React.

EDUCATION

Chechen Industrial College, Grozny — diploma, Computer Operator (Electronic Computing Machines), 2017

2017–2019 — worked while training as a programmer and doing a developer internship; studied Java Core, Spring and SQL on my own.

LANGUAGES

- **Russian** — native.
- **English** — solid B1: I speak and write it, read documentation and source code without a dictionary, and handle written communication and code review in English.