Stjepan Mamuša

Backend software engineer

Split, Croatia · Remote (CET)
mamusa.stjepan@borac.dev github.com/stjepanmamusa linkedin.com/in/stjepanmamusa

Summary

Backend software engineer with 6+ years building Java and Spring Boot systems across telecom, iGaming and enterprise integration, including production work in the betting industry supporting 70,000+ concurrent users during peak sportsbook traffic. I specialise in modernising legacy systems while keeping their business logic intact, most recently rewriting a SOAP/Apache Camel integration layer into Spring Boot 3 microservices.

Experience

Software developer, Intellexi — March 2024 – present
Split, Croatia / Remote

- Redesigned client-facing architecture using a BFF pattern (Spring Cloud Gateway), giving 30+ backend microservices a single entry point and centralising Keycloak authentication instead of duplicating it per client.
- Led the rewrite of legacy SOAP/Apache Camel integration services (20+ endpoints each) into Spring Boot 3, improving compatibility with newer services and cutting new-developer ramp-up time previously lost to learning the old Camel route logic.
- Built load-testing scripts that surfaced critical bottlenecks in post-login customer data and contract-detail retrieval, providing the evidence that drove downstream query optimisation.
- Delivered secure REST adaptation layers for banking integrations across Docker, PostgreSQL, AWS, OpenShift and 3scale.

Software developer, Fortuna Entertainment Group (iGaming) — March 2022 – March 2024

- Rebuilt a fragile, single-stage CI pipeline prone to unexplained failures into a staged, flag-driven Jenkins pipeline (build → functional tests → integration tests → deploy) across 5 services, restoring visibility and reliability to releases.
- Owned the authentication server underpinning every login on the platform, keeping it stable through peaks of 70,000+ concurrent users during major sportsbook events.
- Diagnosed a CPU-critical bug in session-tracking metrics under peak load and replaced faulty counter logic with a set-based unique-user approach, eliminating the CPU spike for a few MB of added RAM.
- Built unit, integration and E2E test coverage (JUnit 5, Mockito, TestNG, Cucumber) and supported observability via Grafana, Prometheus, AppDynamics and ELK.

Software developer, Ericsson Nikola Tesla d.d. — February 2020 – March 2022

- Designed a one-click Jenkins deployment pipeline (Java + Trilead SSH-based remote CLI) that rolled out a Telegraf monitoring agent across entire data-centre server blocks (2+ servers, 40+ switches, and multiple control/data-plane VMs per block).
- Architected the full monitoring stack (Telegraf, InfluxDB, Grafana) end to end, covering CPU, RAM, disk, file-descriptor and network port metrics across the monitored infrastructure.
- Developed internal engineering tools and test automation in Java and Node.js/Express.js, including regression suites and metrics-report generation APIs.

Early career, 8x8 Communications Cloud · Hrvatski Telekom · Digital Academy/Algebra — 2015 – 2020

- QA and technical support roles in telecom/ICT environments (manual testing, defect analysis, service-quality support); assistant teacher for digital-skills education.

Skills

Backend	Java 11/17/21, Spring Boot, Spring Cloud Gateway, REST API design, Microservices architecture
Security and auth	Keycloak, OWASP practices, SonarQube, API gateway authentication
Testing and quality	JUnit, Mockito, TestNG, Cucumber, Load and performance testing
DevOps and infrastructure	Docker, Jenkins pipeline design, OpenShift / Kubernetes, AWS, CI/CD architecture
Observability	Grafana, Prometheus, InfluxDB, ELK Stack
Data and messaging	PostgreSQL, RabbitMQ
Domain experience	iGaming / betting platforms, Telecom, Banking integrations, High-availability systems

Personal project

CroEx, Croatian bank exchange rates

Compares the exchange rates published by 17 Croatian banks, with each bank's history since 1 January 2023, the euro's first day in Croatia. A Python scraper stores one snapshot per bank per day in PostgreSQL and keeps going when a single bank fails. A read-only Spring Boot API serves the data, and a Next.js frontend calls it from the server only, so the API is never exposed to the browser.

Stack: Python, Playwright, SQLAlchemy, Alembic, PostgreSQL, Java 25, Spring Boot, Next.js, Tailwind, Docker

Education

B.A.Sc. Electronics Engineering (Applied Electronics)

University Department of Professional Studies, Split · `2016 – 2020`

Electrical Engineering & Information Technologies

FESB, University of Split · `2014 – 2016`

Languages

Croatian (native) · English (B2 spoken/written, C1 reading) · German (A1)