Venkat Reddy
Java Full Stack Developer
venkat2022.grs@gmail.com | **+1(774)-708-2689**

PROFESSIONAL SUMMARY:

- Seasoned Java Full Stack Developer with 7+ years of experience delivering scalable, secure, and high-performance enterprise applications across Banking, Financial, and Healthcare domains.
- Proficient in the complete SDLC - from analysis, design, and development to testing, deployment, and production support - within Agile environments.
- Expertise in Java 8/11/17, multi-threading, concurrency, and functional programming (Streams, Lambdas, Optionals) to build high-performance, thread-safe back-end services.
- Skilled in Spring Boot microservices using RESTful APIs, Spring Data JPA, Security, Cloud Config, AOP, and Actuators; experienced in event-driven architectures with Kafka and RabbitMQ.
- Strong front-end expertise with React (Hooks, Redux) and Angular (15, NgRx, RxJS), building responsive SPAs using TypeScript, Tailwind CSS, and modern UI/UX principles.
- Designed secure authentication and authorization flows using OAuth2, JWT, AWS Cognito, and IAM for cross-platform integration between microservices and web applications.
- Proficient in SQL (MySQL, PostgreSQL) and NoSQL (MongoDB, Cassandra, DynamoDB) databases with expertise in query optimization, indexing, Flyway migrations, and Redis caching for high-speed data access.
- Hands-on experience deploying containerized applications using Docker and Kubernetes (EKS/AKS), Helm charts, and Terraform for infrastructure as code and zero-downtime deployments.
- Implemented CI/CD pipelines using Jenkins, GitHub Actions, and Azure DevOps with automated builds, testing, and security scans via SonarQube and Fortify.
- Experienced in building and deploying cloud-native applications using AWS and Azure platforms.
- Strong knowledge of test-driven development (TDD) using JUnit 5, Mockito, REST Assured, Postman, and Cucumber integrated within CI/CD pipelines.
- Adept in Agile methodologies (Scrum, Kanban), code reviews, and cross-functional collaboration, mentoring junior developers and ensuring clean, maintainable code.
- Excellent problem-solving and communication skills with a proven ability to deliver robust, efficient, and secure solutions in fast-paced environments.

Education: Master's degree in Computer Technology from Eastern Illinois University, Illinois, USA.
Certificates: AWS Certified Developer – Associate

TECHNICAL SKILLS:

Programming & Java/J2EE Technologies:	Java 8/11/17, Streams, Lambda Expressions, Collections Framework, Servlets, JSP, JDBC, Multithreading, Concurrency, JavaBeans, Log4j, JUnit 5, TestNG, Mockito, RESTful Web Services, Functional Programming.
Web & Front-End Technologies:	HTML5, CSS3, JavaScript (ES6+), TypeScript, React.js (Hooks, Redux Toolkit), Angular 15 (NgRx, RxJS), Tailwind CSS, AJAX, JSON, Responsive UI/UX Design
Frameworks	Spring 4.x/5.x/Boot 2.x/3.x, Spring Cloud, Spring Security, Spring AOP, Spring Data JPA, Hibernate ORM, Microservices Architecture, JPA, GraphQL API
Build & CI/CD Tools:	Maven, Gradle, Jenkins, GitHub Actions, Azure DevOps, SonarQube, Fortify, REST Assured, Postman, Newman
Version Control Tools:	Git, GitHub, GitLab, Bitbucket, SVN
Databases:	MySQL, PostgreSQL, Oracle, MongoDB, Cassandra, DynamoDB, Redis, Flyway
Cloud & Container Platforms:	AWS (EKS, EC2, S3, RDS, Lambda, DynamoDB, API Gateway, SNS/SQS, CloudFormation, CloudWatch, X-Ray, Cognito), Azure (AKS, App Service, Key Vault, App Config, DevOps Pipelines)
Infrastructure & Orchestration:	Docker, Kubernetes, Helm, Terraform, OpenShift
Messaging & Event Stream	Kafka, RabbitMQ, ActiveMQ
Operating Systems & IDEs:	Windows, macOS, Linux, IntelliJ IDEA, Visual Studio Code, Eclipse
Monitoring &	Prometheus, Grafana, ELK Stack (Elasticsearch, Logstash, Kibana), AWS CloudWatch

Observability:	

Client: JPMorgan Chase | New York, NY **March 2022 – Current Date**
Role: Java Full Stack Developer

Responsibilities:

- Actively participated in all Agile phases — story analysis, design, development, testing, and release — delivering enterprise-grade banking modules with high reliability and faster sprint throughput.
- Designed and implemented Spring Boot microservices using Java 17 and Spring Cloud stack (Eureka, Gateway, Feign, Config Server, and Resilience4j) for service discovery, fault tolerance, and load balancing, replacing outdated Netflix OSS components.
- Developed secure REST and GraphQL APIs using Spring Boot with OAuth2 and JWT authentication, integrated via AWS Cognito and IAM for role-based access and authorization.
- Leveraged Java 17's functional programming features, including Stream API, Optional, and the enhanced Date/Time APIs, to write cleaner and more efficient code.
- Optimized backend performance by developing multithreaded and asynchronous workflows using CompletableFuture, Fork/Join, and parallel streams to process transactions concurrently.
- Designed and implemented event-driven microservices using Apache Kafka with producer/consumer groups, DLQs, and offset management for reliable and high-throughput message processing.
- Utilized Spring Cloud (Config Server, Gateway, Feign, Sleuth, Zipkin, and Resilience4j) to enable seamless service communication and distributed tracing across microservices.
- Contributed to development of asynchronous APIs using Spring Web Flux and Kafka for non-blocking processing under high-load scenarios.
- Integrated Spring Batch for background processing and scheduled financial data reconciliation jobs, improving system efficiency.
- Documented and versioned REST APIs using Open API 3.0 and Swagger UI, improving onboarding and integration with internal and external systems.
- Built resilient microservices using Resilience4j circuit breakers, retry logic, and fallback mechanisms to maintain service uptime during dependency failures.
- Designed a Node.js/Express-based Backend-for-Frontend (BFF) service layer to optimize frontend integration with microservices, reducing payload size by 30% and improving page load times.
- Automated API regression testing by integrating Postman collections and Newman in GitHub Actions pipelines for contract validation and early defect detection.
- Configured API Gateway rate-limiting, throttling, and AWS WAF rules to safeguard REST APIs against DDoS and unauthorized access.
- Developed Angular 15+ single-page applications integrated with REST and GraphQL APIs, building reusable components, dynamic forms, and shared UI modules to support multiple banking features.
- Implemented NgRx for state management along with HTTP interceptors and route guards to handle authentication, centralized error handling, and secure navigation across application modules.
- Built interactive dashboards using Angular and Chart.js, utilizing RxJS Observables for real-time data updates, server-side pagination, and efficient handling of large transaction datasets to improve UI performance.
- Optimized microservice performance through asynchronous processing, Redis-based caching, and SQL query tuning, reducing overall API latency by up to 40% across core banking modules.
- Implemented Flyway-based DB migrations, optimized queries and indexes using RDS Performance Insights, and led MongoDB version upgrades for better analytics performance and scalability.
- Wrote complex SQL and optimized indexes for analytics-heavy queries, reducing DB execution time from seconds to milliseconds in transaction reports.
- Developed and deployed containerized Spring Boot microservices using Docker and Helm on AWS EKS clusters for scalable orchestration.
- Configured and optimized Kubernetes Helm charts for deploying containerized Spring Boot applications, ensuring consistent version control and zero-downtime deployments on EKS.
- Implemented distributed tracing and monitoring using OpenTelemetry integrated with Prometheus, Grafana, X-Ray, and CloudWatch to improve production debugging and system visibility.
- Developed AWS Lambda functions triggered by S3 and DynamoDB streams to handle asynchronous workflows in the credit card fraud detection system, reducing backend load and improving response times.
- Worked with DevOps team to deploy services using Terraform and AWS CDK-based infrastructure configurations.
- Worked with AWS services such as EKS, Lambda, API Gateway, RDS, S3, and CloudWatch as part of building and deploying cloud-native microservices.

- Deployed high-availability Spring Boot REST APIs on AWS EC2 and serverless workflows on AWS Lambda with S3 and API Gateway integration for fault-tolerant, scalable architectures.
- Supported deployment of highly available Spring Boot services using AWS Auto Scaling, ELB, and Multi-AZ RDS configurations.
- Enhanced CI/CD quality gates using SonarQube, PMD, and Checkstyle to improve code quality and maintain deployment standards.
- Achieved over 90%-unit test coverage using JUnit 5 and Mockito with automated validation integrated into CI/CD pipelines, improving code quality and release stability.
- Led peer reviews, mentored 4 junior developers, and enforced clean code standards using GitHub PRs and custom review checklists.
- Managed a trunk-based branching strategy with protected PR rules, and automated multi-module builds using Maven with environment-specific profiles.

Environment: Java 17, Spring (Boot, Cloud & Batch), REST & GraphQL APIs, Angular 15+, NgRx, MySQL, PostgreSQL, MongoDB, AWS (EKS, Lambda, API Gateway, RDS, S3, DynamoDB, CloudWatch, WAF), Docker, Kubernetes, Helm, Terraform, ArgoCD, Jenkins, GitHub Actions, JUnit 5, Mockito, Cypress, Linux.

Client: Barclays Bank | Wilmington, Delaware **April 2019 – November 2021**
Role: Java Full Stack Developer

Responsibilities:
- Actively contributed in Agile sprints, and participated in story grooming, automation discussions, and release planning activities and sprint velocity by 15%.
- Built and modernized backend services using Spring Boot, Spring MVC, Spring Security, and Spring Batch, leveraging dependency injection and modular architecture to create scalable and secure banking applications.
- Contributed to migration of legacy monolithic modules into Spring Boot microservices with REST APIs, centralized logging, and inter-service communication via Feign Clients and Spring Cloud Config.
- Refactored existing Spring MVC modules into lightweight Spring Boot microservices with centralized logging, error handling, and JWT-based authentication, reducing support tickets by 40%.
- Built a token refresh mechanism with OAuth2 and Redis for long-lived sessions in secure banking APIs without degrading performance.
- Designed secure, scalable RESTful APIs using Spring Boot and Spring MVC, integrating OAuth2 and Spring Security for RBAC-based access control.
- Designed and documented REST APIs using Swagger 2.0 and Open API specifications, improving integration and reducing onboarding time for frontend and third-party teams.
- Implemented Spring AOP interceptors for cross-cutting concerns like logging, error handling, and metrics capture to improve performance and maintainability.
- Implemented message-driven architecture using RabbitMQ, ensuring reliable asynchronous processing, improved system decoupling, and scalable event-driven workflows.
- Implemented global HTTP interceptors in React to handle token injection, session management, and centralized error handling across application modules.
- Improved frontend performance by implementing code-splitting and lazy-loaded modules, reducing initial bundle size by 40%.
- Developed and deployed high performing React SPA dashboards with Redux, React Hooks, and async data integration from Spring Boot APIs for real-time banking workflows.
- Built modular React components with lazy loading, memorization, and suspense to boost performance and reduce initial load time by 35% on transaction-heavy views.
- Integrated REST APIs using Redux Thunk and Axios, managing asynchronous data flows, centralized error handling, and improving UI responsiveness across banking dashboards.
- Developed automated test suites with JUnit, Mockito, Jest, and React Testing Library, achieving 85%+ code coverage and ensuring software reliability.
- Automated schema migrations using Flyway for PostgreSQL and MySQL, enabling safe deployments with version-controlled DDL changes across environments.
- Optimized PostgreSQL and MySQL queries through indexing and partitioning, and implemented Spring Batch for ETL and data archival processes, reducing query response times by 50%.
- Integrated MongoDB aggregation pipelines for analytics dashboards, reducing the need for multiple roundtrips and improving frontend load times.

- Supported infrastructure deployments using Azure Resource Manager (ARM) templates in collaboration with DevOps team.
- Worked with platform team to configure Azure API Management policies such as rate limiting and request validation.
- Supported deployment of containerized applications on Azure Kubernetes Service (AKS), configuring health checks and scaling policies.
- Integrated Azure Key Vault and App Config for secure environment-based configurations.
- Modernized monolithic systems by containerizing Spring Boot apps and deploying them on Azure App Services and AKS, cutting infrastructure costs by 20%.
- Enhanced CI/CD pipelines with SonarQube and Fortify SCA for code quality and vulnerability checks before merging to main branches.
- Contributed to CI/CD workflows using Azure DevOps and Jenkins by integrating automated tests and quality checks.
- Containerized applications using Docker and orchestrated deployments with Kubernetes, ensuring high availability, load balancing, and efficient resource utilization.

Environment: Java, Spring Boot, Spring MVC, Spring Security, React, HTML5, CSS3, JavaScript, PostgreSQL, MongoDB, RabbitMQ, Azure DevOps, Docker, Kubernetes, OAuth 2.0, JUnit, Mockito, Jest, React Testing, CI/CD, Agile Scrum.

Client: **Cloud Express Solutions| Bangalore, India** **January 2018 – March 2019**
Role: Java/J2EE Developer

Responsibilities:
- Actively contributed through all SDLC phases - requirement analysis, design, development, testing, and deployment - following Agile Scrum practices with sprint planning and retrospectives.
- Developed modular Java components using OOP principles, Collections API, and robust exception handling to support scalable financial transaction workflows.
- Developed J2EE-based transaction modules using Servlets, JSP, and JDBC, integrated with Spring and Hibernate ORM for efficient data access and persistence.
- Worked on JMS-based asynchronous modules for vendor communication within existing enterprise workflows.
- Implemented Spring Data JPA and Hibernate ORM for database access, optimizing SQL queries using indexing and connection pooling to improve persistence performance.
- Designed and developed RESTful Web Services using Jersey, enabling seamless integration between front-end and back-end systems.
- Developed SOAP-based web services and implemented WS-Security configurations as per enterprise security standards.
- Developed responsive UI components using Angular 6, HTML5, and CSS3, implemented lazy loading and AOT compilation for faster rendering, and integrated them seamlessly with RESTful backend APIs.
- Improved user experience by implementing reactive data handling with RxJS, centralized error management, and dynamic loading indicators for Angular modules.
- Automated database versioning and migrations using Flyway for PostgreSQL, ensuring schema consistency across development and production.
- Worked with DevOps team in configuring AWS RDS (PostgreSQL) environments and supported database connectivity and failover testing.
- Supported deployment of services using AWS CodePipeline and CodeDeploy configured by DevOps team and assisted in Terraform-based environment setup.
- Automated build and deployment workflows using Jenkins and Shell scripts, ensuring consistent CI/CD execution across development and production environment.
- Performed root-cause analysis using application logs from Log4j and AWS CloudWatch, reducing defect resolution time during UAT and production cycles.
- Wrote unit and integration tests using JUnit and Mockito for service and DAO layers, achieving 80%+ test coverage across core modules.
- Collaborated with cross-functional teams - business, QA, and DevOps - to deliver stable releases, validate business logic, and ensure successful production deployments.

Environment: Core Java, J2EE, JSP, JDBC, Hibernate, JPA, HTML, CSS, JavaScript, SOAP, REST (Jersey), MongoDB, AWS (RDS, S3, CodePipeline, CodeDeploy), Terraform, Jenkins, Bitbucket, Postman, Shell Scripts, Agile.