Sai Teja Edikuda

Full Stack Developer

saitejaedikuda7@gmail.com | (913) 214-6833 | Linkedin | Github | Medium | Portfolio

Professional Summary

Highly skilled and experienced Java Full Stack Developer with over 7 years of experience designing, developing, and deploying scalable, high-performance web applications. Proven ability to work across the entire software development lifecycle, from requirements gathering and architecture design to implementation, testing, and deployment. Expertise in a wide range of Java/J2EE technologies, Spring Framework, Spring Boot, RESTful APIs, microservices architectures, and modern front-end frameworks like Angular and React.js. Strong cloud experience with AWS and experience with containerization and orchestration using Docker and Kubernetes. Passionate about building high-quality, secure, and maintainable applications.

Technical Skills

Category	**Technologies**
Programming Languages	Java (8/11/17), C, Java Script, Type Script, SQL, PL/ SQL, Python
Java/J2EE Technologies	Servlets, JSP, JSTL, JDBC, JPA, EJB, JMS, JNDI, JavaBeans, JAXB
Frameworks	Spring MVC, Spring Boot, Spring IOC, Spring AOP, Spring Batch, Spring Security, Spring Data JPA, Spring Cloud, Spring Webflux, Struts, Kafka, Hibernate
Databases	Oracle, SQL Server, MySQL, MongoDB, Cassandra, MariaDB, DynamoDB
Web Technologies	HTML5, CSS3, Bootstrap, SOAP, REST, AJAX, jQuery, JSF
JavaScript Frameworks	Angular (9-13), React.js, Node.js, next.js
Web/App Servers	IBM WebSphere, JBoss, BEA WebLogic, Tomcat
Mark-up/XML Technologies	XML, XSD, XSLT, JSON
Architectural Patterns	Microservices, MVC, SOA
Cloud Technologies	AWS (EC2, RDS, S3, Document DB, ECS, Lambda, CloudWatch, CloudFormation, VPC, IAM, ELB, Route53, SQS, SNS, ES, EKS), OpenShift, Kubernetes, Docker
Testing Tools	JUnit, Mockito, SoapUI, Postman, Selenium WebDriver, TestNG, Cucumber
Logging/Others	Log4j, Splunk, Cloud logging
Version Control	Git, SVN, Bitbucket, GitHub
Development Tools/Environments	Maven, Ant, Eclipse, IntelliJ IDEA, Visual Studio Code, Jira
Middleware/Messaging Systems	Apache Kafka, RabbitMQ, ActiveMQ, JMS.
CI/CD Tools	Jenkins, GitLab CI/CD, Docker Hub, AWS Code Pipeline, AWS Code Build
Additional Tools	Graph QL, RxJS, NgRx, D3.js, Chart.js, Terraform
Design Patterns	MVC, Intercepting Filter, Front Controller, Session Facade, Business Delegate, DTO, DAO, Service Locator.

Professional Experience

Senior Full Stack Developer **May 2023 – Present**

The Huntington Bank, remote

Project Description: As a Senior Full Stack Developer, I contributed to the development and enhancement of the Private Bank Hub, a sophisticated digital banking platform for high-net-worth clients at Huntington Bank. I developed and integrated key services, including a unified financial dashboard, client-advisor matching, goal tracking, financial planning, secure digital storage, predictive analytics for transaction reminders, and a comprehensive investment dashboard. My work involved collaborating with cross-functional teams to elevate the client experience, delivering a user-friendly and comprehensive platform for high-net-worth clients to manage their finances effectively.

Frontend:

- Developed a **dynamic and responsive** client-side application using React and Next.js to provide a seamless user experience.
- Utilized **Redux Toolkit** for efficient and scalable state management, ensuring consistent application states across complex interfaces.
- Implemented **React Query** for efficient asynchronous data fetching and caching, improving application performance and user experience by reducing loading times.
- Designed and built reusable and maintainable UI components with **Material-UI**, ensuring a consistent and polished look across the platform and speeding up development.
- Leveraged **Next.js** for server-side rendering (SSR) and automatic code splitting, enhancing application performance and SEO optimization.
- Created interactive charts and visualizations using **D3.js** and **Chart.js** to provide users with intuitive and insightful data representations for financial analysis and tracking.
- Integrated **WebSockets** for real-time updates and notifications, enhancing interactivity for features such as the investment dashboard and spend analysis.
- Employed **WebRTC** for secure and reliable video call functionality, facilitating direct communication between clients and financial advisors.
- Ensured robustness and reliability of frontend components with comprehensive unit and integration testing using **Jest** and **React Testing Library**, improving code quality and reducing bugs.

Backend:

- Managed the full software development lifecycle (SDLC), from requirement analysis using **UML diagrams** to deployment, ensuring alignment with business objectives and technical requirements.
- Leveraged **Spring Boot** and **Spring Framework** for rapid development and integration of Java EE components, ensuring modularity, scalability, and efficient development processes.
- Designed and implemented efficient **Data Access Objects (DAOs)** and entities using **Hibernate API** and **Hibernate Query Language (HQL)** to optimize database interactions and ensure data integrity.
- Designed and documented **RESTful APIs** using **Spring MVC**, enabling seamless data transmission between frontend and backend services with clear endpoints, data formats (JSON), authentication mechanisms (JWT), and error handling procedures.
- Utilized **Spring Cloud** components (Eureka, Config Server) to build resilient microservices architectures, ensuring high availability, fault tolerance, and dynamic scalability.

- Employed **Spring Batch** for batch processing, efficiently handling large volumes of data through chunk processing, partitioning, and multi-threaded step execution.
- Utilized **PostgreSQL** for relational database management, optimizing complex queries, designing schemas, and implementing stored procedures and triggers for efficient data retrieval and manipulation.
- Integrated **DynamoDB** for NoSQL database solutions, leveraging its scalable, low-latency data storage for handling high-velocity and high-volume data.
- Secured the backend with **Spring Security** by implementing JWT authentication, role-based access control, method-level security annotations, and input validation, adhering to **OWASP best practices** to mitigate security risks.
- Designed and implemented an **event-driven architecture** using **Apache Kafka** to handle real-time events like transactions, account updates, or market changes, ensuring the dashboard reflected the latest information immediately.
- Ensured robustness and reliability of both frontend and backend components through comprehensive **unit, integration, and end-to-end testing** using Jest, React Testing Library, and JUnit, improving code quality and reducing bugs.
- Utilized **Git** for version control, leveraging branching and merging capabilities to manage feature development, bug fixes, and releases. Employed GitHub's pull request functionality for code reviews, ensuring code quality and adherence to coding standards.
- Employed **Docker** for containerization and **Kubernetes** for orchestration, ensuring consistent deployment and management of microservices across different environments.
- Implemented **CI/CD pipelines** using AWS CodePipeline, CodeBuild, and CodeDeploy to automate the build, test, and deployment processes, enhancing development efficiency and reducing deployment errors.
- Utilized **AWS CloudWatch** for monitoring application performance and logging, providing real-time insights and troubleshooting capabilities.

Cloud:

- Deployed the application on **AWS**, leveraging various services for scalability, reliability, and security.
- Utilized **AWS EC2** for compute resources to ensure scalable and reliable application deployment.
- Employed **AWS S3** for secure and scalable storage solutions, managing large volumes of data efficiently.
- Managed relational databases using **AWS RDS** for PostgreSQL and NoSQL databases using DynamoDB, leveraging their managed services for scalability, reliability, and automated backups.
- Used **AWS API Gateway** to manage and scale API endpoints, ensuring secure and efficient access to backend services.

Environment: Java 17, Spring Boot 3, Spring Framework (IoC, annotations), Hibernate API, Hibernate Query Language (HQL), Spring Cloud (Eureka, Config Server), Spring Data JPA, Spring WebFlux, Spring Security 6, PostgreSQL, MySQL (AWS RDS), DynamoDB, Redis, Apache Kafka, React 18, JavaScript (ES6+), Virtual DOM (VDOM), Next.js 12, AWS (EC2, S3, RDS, Lambda, Elastic Beanstalk, CodePipeline, CodeBuild, CodeDeploy, API Gateway, CloudFormation, CloudWatch, CloudTrail, IAM, Fargate), Docker, Kubernetes, WebSockets, WebRTC, Jest, React Testing Library, Redux Toolkit, Axios, Material-UI, IntelliJ IDEA, JIRA, D3.js, Chart.js.

Full Stack Developer **Oct 2021 – Apr 2023**

PwC, remote

Project Description: Contributed as a full stack developer to a Supply Chain Management (SCM), designed on a microservices architecture to optimize logistics, inventory management, and supply chain operations. The suite integrated advanced technologies, including **IoT devices** and **RFID systems**, to enhance visibility and streamline processes from procurement to distribution.

Frontend:

- Developed responsive frontend components using **Angular 13** to interact with RESTful APIs based on a microservices architecture, ensuring seamless communication and data synchronization.
- Implemented **Angular services** and reusable components to encapsulate business logic, promoting modularity and enabling agile updates within the development environment.
- Utilized **RxJS** for managing asynchronous data streams and event handling, ensuring smooth and responsive user interfaces.
- Incorporated **Angular Material** and **Bootstrap** for consistent and responsive UI components, adhering to best practices for design and accessibility.
- Integrated **NgRx** for state management to maintain application state across components, improving performance and maintainability.
- Conducted unit testing using **Jasmine** and **Karma** to ensure code quality and robustness of frontend components.
- Implemented **lazy loading** and **code splitting** to optimize application load times and improve performance.
- Integrated **Server-Sent Events (SSE)** to provide real-time updates on inventory levels, order statuses, and shipment tracking, ensuring up-to-date information for users.

Backend:

- Engineered reactive RESTful API endpoints using **Java**, **Spring Boot**, and **Spring WebFlux** within a microservices framework to ensure high performance and scalability.
- Leveraged **Project Reactor** to build non-blocking and asynchronous data processing pipelines, enhancing the responsiveness and scalability of backend services.
- Employed **functional programming paradigms** with Spring WebFlux to create concise and declarative endpoint handlers.
- Implemented **backpressure mechanisms** using Reactive Streams to gracefully handle varying data loads and prevent resource exhaustion.
- Integrated **service discovery** and **circuit breaker patterns** with Netflix Eureka and Hystrix for enhanced resilience and fault tolerance.
- Integrated distributed tracing and monitoring tools such as **Spring Cloud Sleuth** and **Prometheus** for real-time performance monitoring and troubleshooting of microservices interactions.
- Implemented **OAuth2 authentication** and authorization mechanisms using Spring Security to enhance application security.
- Proactively identified and resolved performance bottlenecks using **JProfiler** and **VisualVM**, optimizing code execution and resource utilization.
- Utilized **PostgreSQL** database with Amazon RDS for managed relational database services, ensuring automated backups, patching, and scalability.
- Utilized **RabbitMQ** for message queuing to ensure reliable and scalable inter-service communication.
- Applied **API Gateway** with Spring Cloud Gateway for request routing, load balancing, and rate limiting, improving security and scalability.
- Implemented **caching strategies** with Redis to enhance performance and reduce latency in data retrieval.

- Developed endpoints using Spring Boot to facilitate **real-time data updates** and one-way communication between the server and clients using **Server-Sent Events (SSE)**, ensuring timely and accurate information flow.
- Ensured robustness and reliability of backend components through comprehensive **unit, integration, and end-to-end testing** using **JUnit 5** and **Mockito**, improving code quality and reducing bugs.
- Employed **Docker** for containerization and **Kubernetes** for orchestration, ensuring consistent deployment and management of microservices across different environments.
- Implemented **CI/CD pipelines** using AWS CodePipeline, CodeBuild, and CodeDeploy to automate the build, test, and deployment processes, enhancing development efficiency and reducing deployment errors.
- Utilized **Git** for version control, leveraging branching and merging capabilities to manage feature development, bug fixes, and releases. Employed GitHub's pull request functionality for code reviews, ensuring code quality and adherence to coding standards.

Cloud:

- Utilized AWS services including **ECS** (Elastic Container Service), **EKS** (Elastic Kubernetes Service), and **CloudWatch** for deploying, managing, and monitoring containerized applications within cloud environments.
- Automated infrastructure provisioning and configuration management using **Terraform** and **AWS CloudFormation**.
- Integrated **AWS RDS** for managed relational database services, providing automated backups, patching, and scalability.
- Applied **AWS Elastic Load Balancing (ELB)** to distribute incoming application traffic across multiple targets for improved fault tolerance and availability.
- Employed **AWS SNS** (Simple Notification Service) for messaging and notification purposes, facilitating communication between services.
- Utilized **AWS Application Load Balancer (ALB)** to support SSE connections, ensuring scalability and reliability for real-time updates.

Environment: Java 11, Spring Framework (including Spring Boot 2.5, Spring Cloud, Spring WebFlux), Hibernate, HTML5, CSS3, TypeScript, Angular 13, RxJS, NgRx, Angular Material, Bootstrap, Jasmine, Karma, Docker, Kubernetes, PostgreSQL 13, AWS (IAM, ECS, EKS, CloudWatch, Lambda, S3, RDS, ELB, SNS), Microservices Architecture, RESTful APIs, Git/GitHub, RabbitMQ, Redis, JUnit 5, Mockito, Spring Batch, Terraform, AWS CloudFormation, API Gateway (Spring Cloud Gateway), Server-Sent Events (SSE), Project Reactor.

Full Stack Developer **Dec 2019 – Sep 2021**

Simmons Bank, Comanche TX

Project Description: The Simmons Bank Digital Banking System transformed customer banking experiences with a digital banking platform. As a Java Full Stack Developer, I contributed to its development, integrating frontend technologies for a user-friendly interface and backend solutions for seamless operations. Key features included secure account management, efficient funds transfers, and innovative functionalities like mobile check deposit and online bill payment. Deployed on AWS, the system ensured robust security and reliability, enhancing overall banking accessibility and customer satisfaction.

Frontend:

- Developed responsive and dynamic user interfaces using **Angular**, **NgRx**, and **SASS**, ensuring a seamless user experience across various devices.
- Implemented a comprehensive **account management dashboard**, enabling users to view real-time balances, transaction histories, and detailed e-statements.
- Integrated real-time alerts and notifications via **WebSockets**, using **NgRx** for alert state management and **Angular Material Snackbar** for notification popups, keeping users informed of important account activities.
- Ensured high code quality and maintainability by incorporating modern JavaScript practices and automated testing with **Jest** and **Enzyme**.
- Employed **Angular Router** for efficient client-side navigation, ensuring smooth transitions between different sections of the application.

Backend:

- Designed and implemented **RESTful APIs** using **Java** and **Spring Boot**, facilitating efficient communication between the frontend and backend systems.
- Leveraged **Hibernate** with **MySQL** for efficient data management, providing fast and reliable access to customer data.
- Utilized **RabbitMQ** to handle real-time events such as transactions and account updates, ensuring the system reflects the latest information accurately and promptly.
- Used **JUnit 5** and **Mockito 3** for unit and integration testing, maintaining high code quality and reliability.
- Implemented a robust **authentication and authorization framework** using **Spring Security** and **OpenID Connect (OIDC)**, ensuring secure access to sensitive financial information and enabling single sign-on (SSO) capabilities for a seamless user experience.
- Leveraged **OIDC's ID Tokens** and **UserInfo endpoints** to verify user identities and obtain essential profile attributes, enhancing security and personalization.
- Employed **Spring Cloud Config** for centralized configuration management, facilitating consistent environment settings across microservices.
- Leveraged **Docker** for containerizing backend services, ensuring consistency across development and production environments.
- Implemented robust input validation mechanisms using **Bean Validation** and custom validators, mitigating risks like **SQL injection** and **XSS attacks**, and enhancing overall system security.

Cloud:

- Deployed and managed **Spring Boot-based backend services** on **AWS Elastic Beanstalk**, automating scaling and streamlining the deployment process for enhanced efficiency.
- Provisioned **Amazon RDS** for managing data, ensuring high availability, scalability, and performance.
- Secured data access with **AWS IAM** and protected against web attacks using **AWS WAF**, adhering to industry security standards and best practices.
- Employed **AWS X-Ray** to trace microservice requests, optimizing application performance, and identifying bottlenecks in the system.
- Utilized **AWS CodePipeline** and **CodeDeploy** for continuous integration and deployment, ensuring rapid and reliable updates to the application.

Environment: Angular 9, NgRx 10, CSS3, HTML5, JavaScript, Java 11, Spring Boot 2.3, Hibernate 5.4, Apache Kafka, WebSockets, AWS (Elastic Beanstalk, EC2, IAM, Lambda, CloudFront, CloudWatch, DynamoDB, S3, CodePipeline, CodeDeploy), Jenkins, OAuth2.0, JUnit 5, Mockito 3, D3.js, Chart.js, Docker, Spring Security 5.3, Spring Cloud Config, Angular Router, Jest 26, Enzyme 3.

Junior Web Developer **July 2017 – Nov 2019**

Tvisha Technologies Pvt Ltd, Hyderabad

Project Description: As a Java web developer, I contributed to the development of a web application for booking movie tickets. This project marked a significant learning curve as it was my first professional experience. I was involved in creating a scalable and high-performance solution using various technologies and frameworks.

Frontend:

- Developed the user interface using **JSF**, **JSP**, **HTML5**, **CSS3**, and **JavaScript**, ensuring a **responsive and user-friendly** experience.
- Utilized **Bootstrap 3** for creating a **responsive design**, ensuring compatibility across different devices and screen sizes.
- Enhanced user interface with **AJAX** for **asynchronous data loading** and improved responsiveness.
- Integrated third-party libraries like **jQuery** for **enhanced DOM manipulation** and event handling.
- Implemented client-side form validation using **JavaScript** and **HTML5 features** to improve data input accuracy.

Backend:

- Assisted in developing web applications using **Java** and **Spring MVC**, contributing to the creation of **scalable and high-performance** solutions.
- Implemented various design patterns such as **Singleton**, **Factory**, and **J2EE patterns** like **Business Delegate**, **Session Facade**, **Value Object**, and **DAO**.
- Utilized the Spring Framework's **IoC (Inversion of Control) Dependency Injection** to inject service objects into action classes using the **Service Locator Design Pattern**.
- Developed necessary parsing and XML construction logic using **JAXB**.
- Utilized **Java Persistence API (JPA)** and **Hibernate** for **database transactions**, ensuring efficient data management and retrieval.
- Participated in **database schema design** and **query optimization**, working with **Oracle 12c** to ensure efficient data storage and retrieval.
- Implemented validation and error-handling mechanisms to ensure **data integrity** and prevent security vulnerabilities such as **SQL injection** and **cross-site scripting (XSS)**.

Cloud:

- Utilized **Git** for version control and code management, facilitating **collaborative development** and ensuring code integrity and traceability.
- Worked on **bug fixes** and **feature enhancements** based on user feedback, ensuring continuous improvement and user satisfaction.
- Applied **Agile methodologies** and participated in daily stand-ups and sprint reviews, contributing to continuous improvement and efficient project delivery.
- Used **Log4j** and **SLF4J** for logging, capturing logs that include runtime exceptions to aid in debugging and monitoring.
- Utilized **Jenkins** for **continuous integration and continuous deployment (CI/CD)**, ensuring smooth and automated delivery of updates.

Environment: Java 8, Spring MVC 5, JSF 2.2, JSP, HTML5, CSS3, Bootstrap 3, JavaScript, Hibernate 5, Design Patterns, XML, Maven, Oracle 12c, Git, Log4j 2, SLF4J, JUnit 4, Mockito 2, Jenkins, AJAX, JSON.

Education

Masters in Computer Science -
University of Central Missouri