**Name: Lokesh Mudduluru**

**Ph.no: +1872-292-0760**

**Gmail.ID: lokeshm4575@gmail.com**

**LinkedIn:** [**linkedin.com/in/mudduluru-lokesh**](https://www.linkedin.com/in/mudduluru-lokesh)

**Professional Summary:**

* Front-End React Developer with over 10 years of comprehensive expertise in the Software Development Life Cycle (SDLC). Demonstrated proficiency in building, testing, and maintaining cutting-edge web applications within Cloud-based environments. Adept at translating UI/UX designs into seamless, pixel-perfect front-end code that delivers outstanding user experiences across a broad range of devices and browsers.
* Expert in modern front-end technologies including React.js, Next.js, JavaScript (ES6+), and TypeScript, focusing on delivering responsive and SEO-friendly web applications.
* Deep understanding of AWS SQS for handling message processing errors, retry policies, and filtering policies to optimize message delivery and efficiency in React applications.
* Skilled in state management with libraries like Redux and MobX, ensuring smooth data flow, efficient state management, and scalable architecture.
* Proven ability to build progressive web apps (PWA) and single-page applications (SPAs) using React.js, implementing advanced server-side rendering (SSR) and static site generation (SSG) features with Next.js.
* Strong command over JavaScript ES6+ features, as well as proficiency in modern development tools like Webpack, Babel, and ESLint for enhanced performance, compatibility, and maintainability.
* Extensive experience with RESTful APIs, GraphQL, and asynchronous programming to integrate and manage backend data, using tools like Axios and Fetch API.
* Well-versed in responsive web design, utilizing CSS frameworks and principles to ensure a fluid experience across desktop, tablet, and mobile platforms.
* Experienced in utilizing Jest, React Testing Library, and Enzyme for robust UI testing, with strong knowledge of continuous integration/continuous deployment (CI/CD) pipelines.
* Strong understanding of version control systems like Git and collaboration platforms such as GitHub and Bitbucket for team-based development and code management.

**Technical Skills:**

|  |  |
| --- | --- |
| * **Languages:** | JavaScript, J2EE, SQL, Java |
| * **J2EE Technologies:** | JSP, Servlets, Spring Boot, JDBC, MVC, Java |
| * **Frameworks:** | VueJS, Maven, Spring Boot, AngularJS, ReactJS, Redux, ES6, NodeJS,  MobX |
| * **Web Services:** | RESTful Web Services, Microservices |
| * **Web Technologies:** | HTML5, CSS3, XML, JavaScript, Bootstrap, TypeScript, Angular JS, ReactJS, jQuery, Ajax, JSON, Vue.JS, Prime Ng |
| * **Application/Web Servers:** | XAMPP, Tomcat Server, HTTP Web Server, Apache, Tomcat, Node.JS, JBoss |
| * **Persistence Layer:** | Hibernate, JPA |
| * **Task Runner:** | Grunt, Gulp, Bower |
| * **IDEs/Tools:** | Spring Tool Suite, Eclipse, SourceTree, MS Visual Studio, Adobe FLEX |
| * **Web Tools:** | HTML5, CSS3, JavaScript, TypeScript, JSTL, jQuery, Ajax |
| * **Version Control:** | GIT, SourceTree |
| * **Database:** | Oracle 11g, MySQL Workbench, SQL Server, PostgreSQL, DynamoDB, MongoDB |
| * **Cloud Tools:** | AWS, GCP, Azure |
| * **Operating System:** | Windows, Linux, macOS |
| * **Testing Frameworks:** | Jasmine, Karma, Protractor for E2E, EmberJS, Junit |
| * **Documentation Tools:** | MS Office, Visio, Confluence, GitLab |
| * **Browser:** | IE, Firefox, Opera, Chrome, Safari |
| * **Tools:** | SoapUI, Postman, Jira, Confluence, Bamboo, Kubernetes, Bitbucket, Rest Client, Splunk, SonarQube, GitLab |

**Professional Experience:**

**Client: TIAA, TX Aug 2023 – Present**

**Role: Sr. Front End React Developer**

**Responsibilities:**

* Implemented dynamic features using ReactJS to enhance real-time user interaction within the cloud application interface.
* Integrated RESTful web services and microservices into the UI, fostering efficient communication between front-end and back-end components in the cloud.
* Utilized AWS SNS as a publish/subscribe messaging system to broadcast messages to multiple subscribers, enabling efficient and scalable communication between React components and backend services.
* Utilized Middleware, Redux-thunk, and react-redux for asynchronous data fetching and communication with RESTful web services.
* Designed and implemented data visualization components such as charts, graphs, and dashboards using React libraries like D3.js or Chart.js, presenting complex data in an intuitive and visually appealing manner.
* Built SPAs using React Router for client-side routing, providing seamless navigation and a smooth user experience without page reloads.
* Implemented AWS Simple Notification Service (SNS) to facilitate event-driven communication between microservices and React UI components, enabling real-time updates and notifications.
* Implemented Jest as the primary testing framework for React applications, creating unit tests, integration tests, and snapshot tests to cover a wide range of UI functionalities.
* Developed React applications integrated with GraphQL to efficiently fetch and manage data from multiple sources, ensuring optimal performance and responsiveness.
* Implemented dead letter queues in AWS SQS to capture and handle failed messages, facilitating troubleshooting and error resolution in React UI components and backend services.
* Utilized Apollo Client to seamlessly integrate GraphQL queries, mutations, and subscriptions into React components, facilitating real-time data updates and seamless user experiences.
* Developed Jest test suites to ensure comprehensive code coverage, identifying edge cases and potential bugs early in the development cycle. **Implemented React performance optimizations** using techniques like **memoization (React.memo)** and **React.lazy** with **Suspense**, which reduced unnecessary re-rendering and improved the app's performance, especially for large-scale components.
* Applied advanced **React Hooks**, including **useReducer** for complex state management and **useMemo/useCallback** to enhance performance by reducing re-calculations and re-renders.
* **Optimized bundle size** and **improved page load times** using **React.lazy** and **dynamic imports** alongside **code splitting** strategies.
* Integrated **Cypress** for **End-to-End (E2E) testing** to ensure the user flows worked across different pages and actions, identifying regressions earlier in the development cycle.
* Employed **React Query** for efficient **data fetching, caching, and synchronization**, simplifying the handling of server state while improving performance and reducing redundant API calls.
* Utilized **Core Web Vitals** to measure and optimize performance metrics, focusing on **First Contentful Paint (FCP)** and **Largest Contentful Paint (LCP)** to improve user experience and meet **SEO** performance standards.
* Leveraged **Netlify** for seamless deployment of React applications, optimizing build pipelines for continuous integration and faster updates to production.
* Utilized Scrum artifacts such as burndown charts, sprint backlogs, and task boards to monitor progress, track team velocity, and identify potential bottlenecks for timely resolution.
* Worked closely with cross-functional teams, including backend developers, designers, and QA engineers, to deliver high-quality React UI components and features within scheduled sprint timelines.
* Created responsive layouts with React and CSS frameworks like Bootstrap or Material-UI, ensuring cross-browser compatibility and mobile responsiveness. Designed Redux reducers to handle actions dispatched by UI components, facilitating seamless state updates and enabling better separation of concerns.
* Integrated AWS DynamoDB for storing and managing NoSQL data in frontend applications, ensuring high availability and low latency access to data. Developed Single single-page applications (SPAs) using modern frontend frameworks such as React to create seamless user experiences with fluid navigation and dynamic content loading. Integrated React frontend with backend APIs using technologies like RESTful APIs, GraphQL, or WebSocket for real-time data exchange, enabling rich and interactive user experiences.
* Utilized state management libraries like Redux, MobX, or Context API in React applications to manage application state efficiently, ensuring data consistency and synchronization across components.
* Implemented client-side routing using libraries like React Router, Vue Router to enable navigation within the application without full page reloads, enhancing performance and user engagement
* Implemented robust security measures within the UI, addressing potential vulnerabilities and ensuring data integrity in the cloud environment.
* Utilized AWS CloudWatch for monitoring and logging frontend applications, enabling real-time visibility into application performance and behavior.
* Developed complex compositions consisting of many dynamic components using the React.js library. Utilized ReactJS Virtual DOM, React views, incorporating JSX for rendering components with custom HTML directives.

**Client: Caremore Health, NC Feb 2021 – July 2023**

**Role: Sr.Front End React Developer**

**Responsibilities:**

* Developed progressive web app (PWA) features such as offline support, push notifications, and add-to-home screen functionality, leveraging service workers and web app manifests to enhance accessibility and engagement.
* Implemented container components connected to the Redux store, passing down state and dispatch functions as props to presentational components, promoting a clear separation of concerns between UI and logic.
* Utilized Redux DevTools to debug and monitor state changes in real-time during development, improving the efficiency of troubleshooting and optimization efforts.
* Utilized Next.js's automatic code splitting feature to optimize bundle sizes and improve page load times, enhancing overall application performance.
* Developed SPA using React JS to utilize the HTML5 DOM features and update elements that are modified in the webpage. Implemented React.memo and useMemo to reduce unnecessary renders, especially for performance-heavy components like dynamic tables and data grids.
* Managed complex side-effects and state with useReducer and custom hooks, streamlining business logic and improving the maintainability of the React codebase.
* Added support for TypeScript's advanced types, ensuring type safety across the entire React project, including the use of interfaces and generics for reusable components and GraphQL queries.
* Enhanced testing by integrating React Testing Library for more intuitive and maintainable tests, complementing Jest for unit and snapshot testing of React components.
* Improved accessibility by using ARIA attributes and performing audits with axe to ensure compliance with WCAG standards, making the app accessible for a wider audience.
* Implemented React Query for managing server state, allowing smoother data fetching from GraphQL endpoints while reducing loading times and improving cache management.
* Utilized Vercel for deploying Next.js applications, leveraging its automatic scaling and serverless functions to optimize both performance and infrastructure costs.
* Developed dynamic and interactive web applications using core JavaScript, enhancing user experiences with real-time updates and rich functionalities.
* Implemented comprehensive unit tests using Jest to ensure the integrity of React components , application logic.
* Utilized WebAssembly to offload computationally intensive tasks from JavaScript, significantly improving the performance of React applications, especially for complex calculations and data processing.
* Collaborated closely with backend developers to design and implement GraphQL schemas and resolvers, ensuring data consistency and integrity throughout the application.
* Integrated Next.js with React components and libraries, utilizing its built-in support for React features such as hooks, context API, and state management for efficient development workflows.
* Utilized Jest's mocking capabilities to isolate components, simulate different scenarios, and test component behavior accurately in controlled environments.
* Utilized AWS Amplify DataStore to enable offline data access and synchronization in React applications, providing a seamless user experience even in low or no connectivity scenarios.
* Implemented pagination, filtering, and sorting functionalities using GraphQL queries to optimize data retrieval and enhance user experience in large-scale applications.
* Utilized JavaScript for direct manipulation of the DOM to create dynamic content, animations, and interactive elements, optimizing the performance and responsiveness of web pages.
* Integrated Jest tests into continuous integration (CI) pipelines, automating test execution and providing rapid feedback on code changes to maintain code quality and stability.
* Developed mock Api’s to test the framework independency.
* Implemented caching strategies and fine-tuned DynamoDB queries to minimize latency and improve React application performance, ensuring rapid data retrieval and rendering for enhanced user experience.
* Developed scalable and performant web applications using Next.js, leveraging its server-side rendering (SSR) and static site generation (SSG) capabilities for optimized user experiences.
* Used React JS with Redux to separate the data and presentation layer. Utilized Redux stores to handle the states and update the UI.
* Used react-data grid to populate the dynamic data within the table. Used react-bootstrap and SASS for handling responsiveness in the application and parse correct styles on prop changes. Resolved Cross Browser Compatible Issues for different versions of IE, Safari, Chrome, Firefox, Opera.
* Implemented client-side form validation using libraries like Formik, Yup, or Vuelidate to enhance data integrity and provide instant feedback to users, improving usability and reducing errors.
* Optimized SPA performance through techniques like code splitting, bundle optimization, and caching strategies to minimize load times and improve responsiveness, ensuring a smooth and fast user experience.
* Implemented dynamic routing in Next.js to create SEO-friendly and easily navigable web applications with clean and intuitive URL structures.

**Client: US Bank, La Quinta, CA July 2019 – Jan 2021**

**Role: Front-End React Developer**

**Responsibilities:**

* Developed responsive and adaptive layouts using CSS frameworks like Bootstrap, Material UI, or Tailwind CSS, to ensure optimal user experience across various devices and screen sizes.
* Designed and implemented actions creators to encapsulate action logic and promote code reusability across different parts of the application, enhancing maintainability and readability.
* Leveraged Redux middleware for features like caching, logging, or authentication, extending the capabilities of Redux and enhancing the overall user experience.
* Implemented authentication and authorization mechanisms using JWT (JSON Web Tokens) and OAuth with Node.js to secure React applications.
* Optimized state management and data flow within React applications using ES6 syntax, including rest/spread operators and default parameters, to handle dynamic data more efficiently.
* Employed JavaScript’s asynchronous programming capabilities (Promises, async/await) to manage API calls and data fetching, ensuring smooth and non-blocking user experiences.
* Developed reusable and maintainable React components using TypeScript, ensuring clear and robust type interfaces. Experienced in developing single-page applications and component-based architecture using Vue.js.
* Integrated Redux with React Router for managing application navigation and state, ensuring consistent routing behavior and synchronizing UI with application state changes.
* Re-architected the existing framework in Vue.JS, optimizing component reusability and implementing webpack configuration for sass/css. Utilized TypeScript’s powerful type system to create generic components, enhancing reusability and reducing redundancy.
* Implemented a robust component-based architecture in ReactJS and Redux, optimizing code reusability and maintainability. Developed front-end components to consume RESTful services provided by Spring Boot, ensuring efficient and secure data retrieval and manipulation.
* Developed server-side logic and APIs using Node.js and Express.js to support dynamic data fetching and processing in React applications. Experienced in collaborating with cross-functional teams, following coding standards, and adhering to best practices for Vue.js development.
* Built real-time features and functionality in React applications using WebSocket libraries like Socket.io with Node.js for bidirectional communication. Employed React-router-dom for seamless navigation and configuration of components within the application.
* Collaborated with Node Js Developers to construct REST APIs for modules like user, products, and checkout, enhancing overall functionality.
* Experience in building and deploying Vue.js applications, including optimizing performance and leveraging server-side rendering (SSR).
* Employed Charles Proxy to inspect HTTP/HTTPS requests and responses, gaining insights into data exchanges.
* Maintained states in stores and dispatched actions using Redux, implementing dynamic functionality through Action Creators.
* Implemented strict null checks and optional chaining in TypeScript to handle nullable values and prevent common runtime errors like null pointer exceptions.
* Integrated third-party APIs and services into React applications using Node.js libraries like Axios and Fetch for data retrieval and manipulation.

**Client: Happiest Minds Software Pvt Ltd, Hyderabad, India Dec 2017 – Mar 2019**

**Role: Front End Developer**

**Responsibilities:**

* Developed and maintained frontend applications using Java and related technologies such as JSP, Servlets, and JSTL. Deployed web applications on Apache Tomcat, AWS, and GCP, enhancing scalability and performance.
* Integrated RESTful web services in Java applications to interact with backend systems and provide data to the front end. Maintained version control using Git throughout the entire development cycle.
* Implemented React.js to build interactive and dynamic user interfaces for web applications, utilizing components for modular and reusable UI development.
* Utilized React Router for client-side routing, enabling navigation between different views and pages in single-page applications (SPAs) built with React.
* Developed custom Node.js scripts and utilities for automating repetitive tasks, such as data migration, database seeding, and image optimization in React projects.
* Implemented GraphQL APIs with Apollo Server and Node.js to efficiently fetch and manage data for React components, improving data fetching flexibility and performance.
* Implemented Continuous Deployment using Jenkins, adhering to Test-Driven Development (TDD) principles.
* Supported the backend team in handling JSON data from the MongoDB database. Developed dynamic and interactive user interfaces with React, utilizing JSX syntax and component-based architecture to create modular and maintainable UIs. Developed Node JS applications, integrating backend with front-end ReactJS services.
* Developed and maintained cross-platform mobile applications using React Native, ensuring consistent performance and UI/UX across both iOS and Android devices.
* Developed reusable React components using ES6 classes and inheritance, adhering to object-oriented programming principles, which improved code reusability across multiple projects.
* Stayed current with emerging trends in cloud computing, implementing innovative solutions to enhance the cloud application’s capabilities.
* Utilized Node.js for implementing server-side caching mechanisms, such as Redis or Memcached, to optimize performance and reduce latency in React applications.
* Utilized ReactJS and Angular to create modular and scalable UI components for the cloud-based application. Implemented Single Page Application (SPA) architecture using Angular for seamless navigation and improved user experience. Utilized Node.js as a proxy for RESTful service interaction and interfaced with PostgreSQL Database. Used version control systems like Git and collaboration platforms like GitHub or GitLab for code management, branching, merging, and collaborative development workflows.
* Integrated React applications into CI/CD pipelines using tools like Jenkins, CircleCI, or GitHub Actions, automating testing, building, and deployment processes for efficient and reliable software delivery.
* Collaborated with backend developers to integrate cloud-based RESTful APIs, ensuring efficient data flow between UI and backend services.

**Client: Sonata Software Pvt Ltd, Hyderabad, India May 2014 – Nov 2017**

**Role: Java Front-End Developer**

**Responsibilities:**

* Developed custom Java solutions for handling user authentication, authorization, and session management in web applications. Coded JavaScript for page functionality and Pop-up Screens.
* Employed design patterns and best practices to ensure maintainability and scalability of the UI codebase in a cloud environment. Developed controllers in AngularJS to manage the application's logic, interact with the view.
* Developed modular and scalable UI components using TypeScript and React, enhancing code maintainability and readability while ensuring type safety and robustness.
* Utilized AngularJS services to encapsulate and share business logic, facilitating code reusability.
* Implemented Continuous Integration/Continuous Deployment (CI/CD) pipelines for the cloud-based UI, ensuring efficient and reliable releases. Developed data insertion forms and validated them using JavaScript.
* Used JavaScript and XML to update a portion of a web page thus reducing bandwidth usage and load time and add modal dialog in web pages to get user input and requests.
* Implemented reusable, modular components in React Native to streamline development and maintain consistency across the application, reducing code duplication.
* Implemented TypeScript interfaces and types to define clear contracts between components, APIs, and external libraries, reducing runtime errors and improving code reliability.
* Used Java build tools like Maven or Gradle for project management and dependency management.
* Created custom directives in AngularJS to encapsulate reusable components and enhance code modularity.
* Integrated Spring Boot back-end APIs with front-end applications, enabling seamless data exchange and dynamic content rendering in the user interface. Integrated TypeScript with popular React frameworks and libraries such as Redux, Material-UI, and Styled Components, ensuring seamless interoperability and enhanced developer experience. Utilized new software methodologies to be able to adapt to changes in requirements quickly
* Ensured that all components of the application are platform and browser compatible.
* Used JavaScript DOM manipulation and JavaScript event to generate the data result in UI.