Building SPA using React

Phase	Scope	Week	Coverage	Demonstrable Outcomes
UI & Front-end, React	Build & Test Modern JavaScript (ES6) programs	1	• Git and Engineering Environment (VSCode) • HTML5, CSS3, Blocks, Floats and Layouts, MediaQueries for Responsive Web Design • Modern JavaScript, Basics of NodeJS (NPM packages and modules) • ES6 (Closures, Async, Call backs), Promises, Async and Await • Browser Debugging Tools • Performance Test using Lighthouse • Linting using ESLint	• Working VSCode • Ability to create basic HTML design, ES6 (JS & TS) • Ability to work with closures, async & await, callbacks • Ability to create and apply style sheets • Ability to create Class components (OOPS concepts for Typescript) • Understand Virtual DOM • Working with NPM packages and modules • Best coding, error handling practices • Prettier, Beautify, Code Indentation Tools • Working with VSCODE, ESLINT • Ability to debug applications specifically using browsers (using toolbar) • Ability to do performance test using Lighthouse
	Working with Advanced JS & Typescript & Introduction to React	2	• DOM Manipulation, Cross- browser JavaScript problems, DOM elements, loops, stack trace for a function • Error Handling, Loading and Runtime Errors, Type Error • Unit Testing using Mocha & Chai, • Introduction to JSON • Typescript – OO Concepts, Basic Types, Interfaces, Functions, Literal Types, Unions and Modules. • ***Capstone Project Start*** • Building SPAs • Interactive Web pages with React JS • Component Life cycle, virtual DOM, JSX Pages	
	Basic Single Page Applications Using React & Redux	3-4	• States and Props, Webpack • React Routers, Rendering Elements, • Functional Components, State and Lifecycle, Handling Events, Conditional Rendering, Lists and Keys, Forms Lifting • State Up, React-hooks & Context • Integrating APIs with Fetch/ Axios – GET, POST,PUT,DELETE • Pure Components • Unit Testing ReactJS Apps using JEST/Enzyme • Mocking SSL certificate	• Bundling using Webpack and optimization • Ability to Create React JSX • Good understanding of Life Cycle Methods • Ability to use Form Validation and data validation • Ability to create functional components • Ability to use context API (Interaction with DOM components- passing data between components) • Ability to call external API's (Async calls and Middleware with redux and Axios) • Ability to create unit test cases using Jest/Enzyme/RTL • Mocking Window SSL certificate

			• Redux Data, Async and Data Fetching, • Hook rules • RxJS & Redux- Observables, Reducers & Actions, Implement Undo History, Immutable JS • Working with Localization and Internationalization (i18n) • Performance and Normalizing Data, Asynchronous Operations, Memorization Redux-Thunk; Redux-Saga • JWT authentication, Session Cookies	• Ability to utilize state management libraries like redux, mbax - (Store/actions/reducers) • Working with Authentication and Role Based Access including JWT + Session Cookies • Ability to use middleware's - Redux thunk / SAGA • Ability to work with Localization and Internationalization (i18n) • Best coding, error handling practices
Capstone Project - Apply & Demonstrate competency gained			• Combining UI/UX, JavaScript and React to Integrate & Re-factor to complete the web application	