React JS Course Content

Introduction to React JS

In this module, you will learn about ReactJS - an open-source JavaScript Library. This react.js is maintained by Facebook and mainly used for building user interfaces. The prominent feature of this JavaScript library is that it plays an important role in developing mobile or single-page applications. Moreover, in this section, you will get through various components of React, the importance of DOM, and the concept of Nesting. Below you will find a list of key concepts of ReactJS.

- An Introduction to Web Application Development – Basic building blocks of Web Application Development. – HTML-CSS-JS
- Real World SPAs and React Web Apps.
- An Overview of ReactJS.
- Understanding Single Page Apps and Multi-Page Applications.
- What is a DOM, VirtualDOM, and ShadowDOM?
- Installing ReactJS and writing our first Hello World code.
- The Folder Structure of React Application.
- Basic Building Blocks of React JS – (components,state,props,jsx)
- What are the components? Understanding the component basics and different types of components.
- What are the functional components and container components?
- Creating your first class-based component and functional component.
- Understanding JSX and JSX restrictions.
- Creating more components and using them.
- Component Nesting.
- What are props and a simple understanding of props?
- Passing Props into components.
- Accessing props inside the components.

Creating reusable dynamic components

Learning Outcome:

You gain basic knowledge about ReactJS by the end of this section. You will also be in a position to build different user interfaces using React.

React UI Frameworks and Styling Components in React JS

In this section, you will come across all the UI frameworks of React along with their usage, features, importance, and popularity. These frameworks are built using a set of React Components for easier and faster web development. You will also learn about the usage of

stylish components in ReactJS. You will get through some of the key essentials of frameworks and components below

- Styling React components. The different ways of styling the react components.
- Inline styles and external styles to the components.
- Styling patterns in React Components – using sass
- Using styled-components.
- Introduction to ReactJS UI frameworks – react bootstrap, reactstrap,react-semantic UI , material-ui etc.
- Integrating and using the above UI frameworks in ReactJS.
- Creating a navigation-bar in reactJS using the above frameworks.
- Creating page-views components.
- Creating a reasonable dynamic Jumbotron/banner.
- Introduction to React Router.
- Creating Routes to our Navigation Bar.

Learning Outcome:

By the end of this module, you will gain expertise in using various React components for quick web application development and you will also understand the importance of React UI frameworks.

React JS Architecture

In this module, you will gain deep insights into ReactJS Architecture. React follows Flux architecture for unidirectional data flow. You can clearly understand how the data flows from each React components in the architecture diagram. Additionally, you will also gain expertise in handling events, the importance of two-way binding, props, states, various lifecycle methods, and many more. Below you will find the list of concepts that will be covered in the architecture of ReactJS.

- Advanced Components configuration with the state, props, and children.
- Props and prop-types and default props, Understanding the children property, Understanding and using state, and Differences between props and state
- Handling Events with methods
- Manipulating the state with setState() method
- Difference between stateless and stateful components.
- Passing method references between components.
- Adding two-way binding.
- Rendering content conditionally.
- Making API REQUESTS with React.
- Fetching data – Axios vs Fetch.
- Building Lists of Customers.
- Handling requests with async-await.
- Updating state immutably and after async request.

- Rendering Customers.
- Review of Map Statements.
- Rendering Lists of components.
- The purpose of keys in lists and Implementing keys in lists.
- Handling errors gracefully.
- Introducing Lifecycle methods.
- Why use Lifecycle methods
- Refactoring data loading to Lifecycle methods.
- Showing a loading spinner.
- Handling User Input with forms and events.
- Creating a SearchBar and Event Handlers
- Controlled Elements versus uncontrolled elements.
- Handling forms submitted.
- Understanding this in Javascript and solving context issues.
- Communicating child to parent.
- Invoking callbacks in children.
- Creating a custom Dynamic Input Component.
- Setting up a JS Config for the Form.
- Dynamically create inputs based on JSConfig.
- Adding a Dropdown component.
- Handling user Input of dynamic forms
- Handling form submission(MAKING API CALL USING AXIOS OR FETCH)
- Adding Custom Form Validation.
- Showing validation errors and other error messages.
- Handling overall form validity.

Learning Outcome:

You will gain a clear idea about React.js architecture and its components upon the completion of this module.

Redux and Advanced React Concepts

In this segment, you will gain a clear understanding of the open-source JavaScript library called Redux. It is used along with the most frequently used libraries such as Angular and React for managing the state of the application. The Redux framework has an architecture similar to that of Facebook's Flux architecture and is used for building user interfaces. Furthermore, you will come across advanced react concepts in this section like performing actions for retrieving and passing data, reducer setup, form handling, debugging, and many more. You will get to know more about key highlights of Redux and the advanced concepts of React below

- Using Refs for DOM Access.
- Redux, and The complexity of managing state.
- Understanding the Redux flow.
- Adding Redux to the React Project and Redux Devtools.
- What is the provider and adding the provider?
- Setting Up the Reducer and the Store

- Dispatching the Actions.
- Adding Subscriptions.
- Connecting React to Redux.
- Connecting the store to react.
- Dispatching the Actions from within the Component.
- Passing and Retrieving Data with Actions.
- Switch Case in the reducer
- Updating the State and the Array Immutable.
- Outsourcing ACTION_TYPES
- Combining multiple Reducers.
- Understanding State types.
- Combining Local UI State and Redux.
- Handling Authentication in React.
- Handling Forms using REDUX-FORM.
- What are React Portals and when to use them?
- The Context System with React.
- Replacing Redux with Context or using both together.
- Higher-Order Components in React.
- The concept of Render Props.
- React Debugging
- Error Boundaries.
- React Fragments
- Memoizing in React.
- The concept of Render-Props in React.

Learning Outcome:

By the end of this section, you will gain deeper insights into the Redux framework and its applications. You can manage the state of the applications using Redux easily.

React Hooks(The way of writing code in React in the Future)

Hooks are the latest additional feature in the React 16.11. They provide the best ways to use other React features and state without declaring a class. Additionally, they allow users to share reusable behaviors that are independent of components implementation. Below you will get more key details about React hooks.

- React Hooks – An Introduction, Why React Hooks
- Moving from classes to functional components
- Introducing the first Hook: useState Hook and Second Hook: useEffect Hook
- Data Fetching with hooks/replacing the class lifecycle methods
- Fetching data on component mount with useEffect
- Using async/wait for fetching data in useEffect
- Building custom Hooks
- Hooks under the hood: How hooks work

- Implementing a login form using Hooks and multiple state values.
- Implementing a register form using hooks
- Comparing ways of managing state
- Fetching search results on component update with useEffect
- Fetching data upon submitting the form
- Using the useRef Hook on our search input
- Displaying loading state with useState
- Error Handling and displaying errors with useState
- Using the reducer Pattern
- The useReducer Hook
- Building a complete CRUD application with React HOOKs/ replacing redux
- Using the useContext Hook
- Replacing Redux with the useReducer hook
- Styling customer lists in hooks
- Toggling the CustomerList using hooks
- Adding Customers and Customer Details form using hooks
- Updating the customer details using hooks
- Deleting the customer details using hooks

Learning Outcome:

You will gain expertise on hooks and also you can manipulate the state of the functional components without converting them into class components by the end of this section.