React & Redux – Course Outline

OVERVIEW
React is a JavaScript library for building user interfaces. It is an open source library and is developed by Facebook. It can be used to create both web and mobile apps. React apps are fast, flexible, modular and scalable. React has a robust developer community that's rapidly growing.

Redux is a JavaScript library used to manage an application's front-end state. As single-page apps become more complex, the need for proper state management has become more important than ever. The main benefits of using Redux come in when your app involves shared state. It enables two or more components to share same piece(s) of application state. Redux not only allows for an organized way to store data, it also gives us the ability to quickly obtain that data anywhere within the app.

The goal of this course is to equip you with the skills and experience you'll need to become a professional React developer. It will teach the core knowledge you need to deeply understand, and build React components and structure applications with Redux.

PREREQUISITES
Participants are expected to have experience with building front-end web applications with:
- HTML & CSS
- JavaScript
- Bootstrap (is a plus)

BENEFITS
At the end of this course, the participant will:
- Build amazing single page applications with React & Redux
- Learn and apply fundamental concepts behind structuring Redux applications
- Realize and appreciate the power of building composable components
- Understand and use NPM, Babel and ES6 JavaScript syntax

© Naveen Pete
+91-98864-95913
naveen.shenoy.blr@gmail.com

TOPICS COVERED

1. ES6 & Beyond
    a. Overview
    b. Array helper methods
    c. Const & let
    d. Template literals
    e. Arrow functions
    f. Default function arguments
    g. Rest & Spread
    h. Destructuring – Array & Object
    i. Classes & Inheritance
2. Building Web Applications using React
    a. What is React?
    b. Why React?
    c. Overview of virtual DOM & diffing algorithm
    d. Create-React-App CLI
    e. Introduction to JSX
    f. Components
        i. Stateless (Functional) components
        ii. Stateful (Class) components
        iii. Component Composition
    g. Props
    h. Managing State
        i. Adding state to a component
        ii. Updating state using setState()
    i. Props vs State
    j. Lifecycle methods
    k. Fragments
    l. Forms
        i. Handling events
        ii. Handling lists
        iii. Controlled components
3. React Router
    a. Introduction to SPAs
    b. Why React Router?
    c. Installing React Router
    d. BrowserRouter component
    e. Link component
    f. NavLink component
    g. Route component
    h. Switch component
4. Asynchronous JavaScript
    a. Synchronous vs Asynchronous code
    b. Patterns for Asynchronous code

    c. Callbacks
    d. Callback hell
    e. Promises
    f. async and await
5. Server Communication
    a. Introduction to Fetch API
    b. Making async calls to REST API endpoints using Fetch
6. Introduction to Redux
    a. Why Redux?
    b. Core concepts of Redux
        i. Actions
        ii. Action creators
        iii. Reducers
        iv. The Store
7. React & Redux
    a. The react-redux Node.js package
    b. Provider component
    c. Connecting React components with Redux store
        i. connect() method
        ii. mapStateToProps() method
        iii. mapDispatchToProps() method
    d. Reducer composition – combineReducers() method
    e. Redux Middleware
        i. Redux Thunk
            1. Need for Redux Thunk
            2. Using Thunk to make async calls to REST API endpoints
8. Introduction to Advanced Concepts in React
    a. Context API
    b. React Hooks
    c. Higher Order Components

DAYWISE BREAKUP OF TOPICS

Sl. No.	Topic	Day
1	ES6 & Beyond	1 & 2
2	Building Web Applications using React	1 & 2
3	React Router	3
4	Asynchronous JavaScript	3
4	Server Communication	3
6	Introduction to Redux	3 & 4
7	React & Redux	4
8	Introduction to Advanced Concepts in React	4