REACT JS CURRICULUM

Introduction to React JS

- What is React JS
- Purpose of React JS
- Why do we required React JS
- Features of React JS

React Elements

- What is React Element
- How to create React Element
- Integration of HTML and React
- How to add Inline, Internal and external CSS
- Detailed Understanding About the React.createElement()
- How to create User Interface with React Element

ReactDOM

- What is React DOM
- How to integrate React DOM with HTML
- How ReactDOM is used to add React Element in the DOM
- Understanding of ReactDOM.render()
- Virtual DOM
- How Virtual DOM works

JSX

- What is JSX
- Syntax of JSX
- How JSX is different from HTML
- Rules of JSX
- Integrating babel with Html
- Creating UI with JSX
- Advantages of JSX
- How JSX simplifies Creation of UI in React JS
- What is babel

- Integration of babel with HTML

React Components

- What is React Component
- Advantage of React Component
- How to create React Component
- Types of React Component
- Introduction to Functional Component
- Creating the Functional Component
- Introduction to Class Component
- Creating the class Components
- How components can be used for Reusability

Props

- What is Props
- Purpose of the props
- How to use Props in Functional Component and Class Component
- How to pass Props
- Access the Props
- Pass the different types of data as a props
- Props types

CRA

- What is CRA
- How CRA is used to create Basic React Application
- How to run and stop React Application
- Accessing the React Application
- Understanding the Folder Structure of React Application
- NPM
- What is Node Package Manager
- How to install different Packages usng NPM

Functional Components

- Understanding of Functional Component in Detail
- How to create Functional Component
- Why Functional Components are used over class Components
- How to render the Functional Component
- Sequence of Calling the Functional Component
- Flow of React Application

Class Components

- Understanding of Class Components
- How to create Class Component
- How Class Components are Different Functional Component
- Rendering of the class Component

Static Web Page

- Creating the Static Home Page using Component and Props Concept
- How Components are used to Create User Interface and Reusing the UI
- How Components and props concept can be used reuse the UI

Introduction to Hooks

- Introduction to the Hooks
- Why hooks are introduced
- Rules of using the hooks
- How to import the Hooks and use it
- Listing the important hooks

State and setState

- What is State
- Why do we required state
- How state can be used to create Dynamic User Interface
- Creation of State
- Introduction to first hook useState()
- Understanding of useState()
- How setState() is used and purpose of it

- Understanding in depth of setState and its Working
- Implementing the Counter App
- Implementing the Dynamic Card with Dark and Light Theme
- Implement Theme feature

How to Integrate CSS with React

- How to use Inline Css using style attribute in JSX
- How to integrate External CSS
- Problem with Css
- Using className attribute

Rendering the List using Map()

- What is map() in JS
- How it Works
- Understanding in details about map()
- How map() used to create UI
- Iterating Through Map
- Keys and List.

Axios

- What is axios
- Installing and Integrating Axios with React App
- How to do get() Request with Axios
- Handling the Promise with then and catch
- Handling the Promise with async await

Form Management and Controlled Components

- How to create Form
- Managing the Form using React JS
- onChange event
- Managing the Form using State Concept
- Controlled Form Components

JSON SERVER

- How to create Json Server
- Add the Data in the JSON Server
- Understanding About Client Server Architecture
- Fetching the data From Server using get request
- Understanding of POST, PUT and DELETE Request
- CRUD Operation using JSON Server

Interaction between Components

- Understanding the Relationship between Components
- Parent Child Relation
- Sharing the Data From Parent Component to Child Component using Props
- Sharing Data From Child Component to Parent ®
- Sharing the Data between the Siblings
- Props Drilling
- Problems With Props Drilling
- Introduction to Context

Context API

- Introduction to Context API
- Why Context API
- How Context API solves the Problems of Props Drilling
- Limitation of Context API
- How to Create the Context
- How to access Provider Component
- Understanding or Provider Component
- Storing the Data in Context
- How to make Available the context data to Child Components
- useContext() hook
- Purpose of useContext() hook
- How to access data from context using the useContext() hook

React Routing

- What is Routing
- How to implement routing in React App
- Installing and Configuring the react-router-dom
- BrowserRouter
- Routes

- Route
- Link
- Navigate
- Outlet
- NavLink
- useParams() hook
- useNavigate() hook
- Nested Routing

useRef() hook

- What is useRef() hook
- How it works
- Purpose of the useRef() hook
- Synatx of useRef() hook
- How useRef() hook used to manage the data or store the data
- Difference between useRef() hook and useState() hook
- DOM Manipulation using useRef() hook

DOM Manipulation and UnControlled Components

- How to manipulate the DOM using useRef() hook
- Change the Content of JSX Element
- Change the Style of JSX Element
- Managing the Form using useRef() hook
- What is UnControlled Components

useEffect() hook ®

- What are sideEffects in the React
- Pure functions in JS
- Lifecycle of Components
- Phases of Lifecycle
- What is Mounting and Unmounting
- Mount Phase
- Unmount Phase
- Update Phase
- How useEffect can be used to perform sideEffects in Different Phases of component

- Understanding How useEffect() hook works

Lifecycle Methods ®

- Understanding of Lifecycle Methods in React
- Mount Phase Lifecycle Methods
    - constructor()
    - Static getDerivedStateFromProps()
    - render()
    - componentDidMount()
- Unmount Phase Lifecycle Methods
    - Static getDerivedStateFromProps()
    - shouldComponentUpdate()
    - render()
    - getSnapshotBeforeUpdate()
    - componentDidUpdate()
- Update Phase Lifecycle Methods
    - componentWillUnmount

Error Boundaries ®

- How to manage Fallback UI on Error
- How to implement Error Boundaries
- Static getDerivedStateFromError()
- componentDidCatch()

useReducer() hook

- Understanding of useReducer() hook
- How to manage complex state operation in the reducer
- Reducer
- Dispatch
- Action object
- Types
- Difference between useState() and useReducer() hook

React Profiler ®

- What is React Profiler
- How to implement it

- Purpose of React Profiler

Lazy Loading ®

- What is Lazy Loading
- Benefits of Lazy Loading
- How to Lazy Loading will improve Performance
- Implementation of Lazy Loading

Redux with Functional Component

- What is Redux
- Why Redux
- How redux will help in state management
- Installing and Configuring redux
- Store
- Dispatch
- Reducer
- How to combine Multiple reducers
- Configuring Reducers with Redux Store
- Action
- ActionCreator
- Action Types
- Redux Pattern
- useSelecto() hook
- useDispatch() hook
- Implementing Redux in React Application