Module 9 - Introduction to React.js

1. **What is React.js? How is it different from other JavaScript frameworks and libraries?**

   ANS: React.js is a JavaScript library for building user interfaces. It is use to build dynamic website.

   - React application are built with reusable components, making code easier to maintain and scale.
   - React uses the virtual DOM, which updates only the parts of the webpage that change, making it faster compared to traditional DOM manipulation.
   - Data flows in one direction in react, making it predictable and easier to debug.
   - React can be integrated with other libraries or frameworks and does not enforce strict rules, giving developers more freedom.
   - It works well with tools like Redux, React Router, and other libraries, enhancing its capabilities without being into React itself.
   - React is lightweight, fast, and focuses on building UI components, while many other frameworks handle a broader set of tasks.

2. **Explain the core principles of React such as the virtual DOM and component-based architecture.**

   ANS:

   1. **Virtual DOM**

   - The DOM is like a tree structure that represents your webpage.
   - The virtual DOM is a lighter, faster copy of the DOM that React uses in memory.
   - When something changes in your app, React updates the virtual DOM first, compares it with the real DOM this is called diffing, and then updates only the changed parts in the real DOM.
   - Updating only the parts that changes makes React faster than updating the whole webpage like traditional JavaScript.

   2. **Component based architecture**

   - Components are like small, reusable building blocks of your app. Each component represents a part of the user interface.
   - A React app is built by combining these components together, like building a house with Lego blocks.

   Example:

- Header components for header.
- Footer components for footer.
- productCard components for displaying products.
- Each component is independent and reusable, which makes the code easier to manage and scale.

3. Unidirectional Data Flow

- In React, data flows in one direction, from parent components to child components.
- This means that a parent can pass data to its child through something called props, but the child can not directly modify the parent's data.
- This makes your app's behaviour more predictable and easier to debug.

4. Declarative Programming

- In React, you describe what the UI should look like, and React takes care of the steps to make it happen.
- Instead of manually telling the browser what to do, React handles the details for you.

3. What are the advantages of using React.js in web development?

ANS: It is fast and efficient because it updates only the part that change, making it quick and efficient.

- You can build piece of the user interface and reuse them. Saving time and efforts.
- Easy to learn and use because react has a simple syntax and focuses on one thing-building user interfaces-making it beginner friendly.
- React apps are dynamic. Better user experience.
- In React data flows in unidirectional that means parent component to child component. Easy to track and debug.
- React makes web development faster, easier, and more organized, also enabling great user experience.

JSX (JavaScript XML)

1. What is JSX in React.js? Why is it used?

ANS: JSX stands for JavaScript Extensible Markup Language.

- It is a syntax used in React.js that looks like HTML but is written inside JavaScript.
- You can use JavaScript and HTML together in one place.
- Easier to write and understand the structure of the user interface because it looks like HTML.

- It makes the code more readable.
- Visually separating the user interface from the JavaScript.

2. **How is JSX different from regular JavaScript? Can you write JavaScript inside JSX?**

ANS: JSX looks like HTML, but it is written inside the JavaScript files.

- Regular JavaScript does not look like HTML.
- We have to write JavaScript file into <script> tag. After or before In HTML file.
- JSX is specially designed to create React components, making it simpler to design user interfaces compared to regular JavaScript.
- Yes, we can write JavaScript inside the JSX by using the curly braces "{}".
- You can also use functions, variables, calculations, or conditions inside the curly braces {}.
- JSX makes it easy combine HTML-like code and JavaScript.

3. **Discuss the importance of using curly braces {} in JSX expressions.**

ANS: Curly braces are very important in JSX**.**

- They allow you to add JavaScript logic or values directly inside your JSX code.
- Without them JSX only looks like plain HTML.
- Not able to include any dynamic behaviour.

Importance:

- To add dynamic content. Like display JavaScript variables, functions, or expressions inside your JSX file.
- We can do calculations or call a function directly.
- We can add conditions to show different content based on conditions.
- We can use map () to render lists or repeated items.

Components(Function & Class Components)

1. **What are components in React? Explain the difference between functional components and class components.**

ANS: Components are like building blocks that let you create reusable pieces of a user interface.

- When creating a React component, the component's name must start with an upper case letter.
- Components can be reused across different parts of the application to maintain the consistency and reduce code duplication.
- React only loads only the necessary components.
- Only the specific components updates of instead of the whole page.

There are two types of React components

1. Functional components

- They are JavaScript function that return a JSX that is user interface structure.
- They are simple and easy to write.
- It is only use return

2. Class Components

- Class components are ES6 classes that extend React.Component.
- They include additional features like state management and lifecycle methods.
- You use this.state and this.setState to manage state.
- It is use render return.

2. How do you pass data to a component using props?

ANS: In React, props is nothing but the properties.

- Props are like arguments you pass to a function.
- They let you send data from parent component to child component.
- In the parent component you pass the data as an attribute.
- In the child component, you access the properties using the props object.
- Props are read only you cannot modify them inside the child component.

They allow the parent component to control what data the child component uses.

3. What is the role of render() in class component?

ANS: The render() in React class component is like the instruction that tell React what to display on the screen.

- Inside the render() method , you return the JSX that defines what should appear on the screen.
- Every class component must have render() methods.

- This render() methods can use this.props and this.state to dynamically display data.

Props and State

1. What are props in React.js? How are props different from state?

ANS: props are pieces of data that a parent component sends to a child component.

- They are like arguments you pass to a function.
- Props cannot be change by the child components.

Props	state
Data passed from parent to a child component.	Data that belongs to a component itself.
Controlled by the parent component.	Controlled by the component itself.
Connot be modified by the child.	Can be updated using setState or useState.
Share data between components.	Manage dynamic, changing data within a component.

2. Explain the concept of state in React and how it is used to manage component data.

ANS: In React, state is a built-in object that stores data or information about a component.

- This data change over time, like when a user clicks a button or types in a form.
- Think of state as the brain of a component, where it keeps track of information that can change over time.
- State holds data that can change. unlike props which are fixed.
- State belongs to the component that defines it. Other components cannot directly modify it.
- When state changes, React automatically re-renders the components to show the updated data.

How state is used in React

- in class components, state is initialized in the constructor.
- In functional components, the useState hook is used.
- Use setState in class components or useState in functional components to update state.

How State Helps Manage Component Data

- State allows a component to store and update data like user inputs, toggle, counters,etc.
- When state changes, the component's UI automatically updates without manual DOM manipulation.
- Each component manages its own state, making it easier to debug and reuse.

3. Why is this.setState() used in class components, and how does it work?

ANS: In React class components, this.setState() is used to update the state of a component.

- When you change the state, you use this.setState() instead of directly modifying it.

Why use this.setState()?

- React needs to know when the state has changed so it can automatically re-render the component to show the updated data.
- If you try to change the state directly, React won't know about the change, and the UI won't update.

How does it works?

- You pass the new state to this.setState(), and React merges it with the existing state.
- After the state is updated, React automatically re-runs the render() methods to update the UI.