**Module-3) React - Components, State, Props**

**3. Components (Functional & Class Components)**

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

In React, components are reusable building blocks of the user interface. Each component is an independent piece of code that returns JSX to describe how the UI should appear. Components can accept props (inputs) and manage their own state.

**There are two types of components in React:**

1. **Functional Components**
   * Defined as simple JavaScript functions.
   * Accept props as arguments and return JSX.
   * With Hooks (e.g., useState, useEffect), they can also handle state and lifecycle.

* **Example:**

function Greeting(props) {

return <h1>Hello, {props.name}!</h1>;

}

1. **Class Components**
   * **Defined using ES6 class syntax and extend React.Component.**
   * **Can hold and manage state using this.state and setState().**
   * **Require a render() method to return JSX.**

* **Example:**

class WelcomeMessage extends React.Component {

render() {

return <h1>Welcome to React!</h1>;

}

}

**Q:-2: How do you pass data to a component using props?**

In React, props (short for properties) are used to pass data from a parent component to a child component. They are read-only and cannot be modified by the child component. Props help make components reusable and dynamic.

**How to pass props:**

1. Pass data from the parent component using attributes.
2. Receive props in the child component and use them inside JSX.

**Q:-3: What is the role of render() in class components?**

In React class components, the **render()** method is a required method that tells React what to display on the screen.

* It returns JSX (or null) which represents the UI of the component.
* React automatically calls the **render()** method whenever the state or props of the component change, so the UI updates accordingly.
* A class component must have a **render()** method, and it can only return one root element.

**4. Props and State**

**Q:-1: What are props in React.js? How are props different from state?**

In React, props (short for *properties*) are used to pass data from a parent component to a child component. They are read-only and help make components dynamic and reusable.

* Props are like function arguments in JavaScript.
* They allow components to display different data without changing the component’s code.

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

In React, state is an object that stores dynamic data within a component. Unlike props, which are passed from parent components, state is managed internally by the component itself.

* State represents information that can change over time (e.g., user input, counter value, toggle status).
* When state changes, React re-renders the component automatically to update the UI.
* State makes components interactive and allows them to respond to user actions or events.

**State in Class Components:**

* Declared inside the constructor using this.state.
* Updated using this.setState().

**State in Functional Components (with Hooks):**

* Managed using the useState Hook.

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

In React class components, this.setState() is the method used to update the component’s state.

* Directly modifying state (e.g., this.state.count = 1) is not allowed because React will not re-render the UI.
* Instead, this.setState() schedules an update to the component’s state object and tells React that this component and its children need to be re-rendered.

**How this.setState() works:**

1. Takes an object or a function as an argument to update state.
2. Merges the new state with the previous state (shallow merge).
3. Triggers React to re-render the component with the updated state.