### ****1. Define Props****

In React, **Props** (short for “properties”) are used to pass data from one component to another, specifically from a parent component to a child component. They are read-only and immutable, meaning a child component cannot modify the props it receives. Props allow components to be dynamic and reusable, enabling the rendering of different data using the same component structure. For example, a UserCard component can display different user information based on the props passed to it.

### ****2. Explain Default Props****

**Default Props** are a way to specify fallback values for props in case they are not provided by the parent component. This is helpful for preventing errors or displaying placeholder values. React allows you to set default props by assigning them to the defaultProps property of the component. For example, if a Greeting component expects a name prop and it is not provided, you can set Greeting.defaultProps = { name: "Guest" }, ensuring the component renders with a default name.

### ****3. Identify the Differences Between State and Props****

While both **State** and **Props** are used to manage and pass data in React, they serve different purposes. **Props** are passed to a component by its parent and are immutable inside the child component, used for configuration or data input. **State**, on the other hand, is managed within the component itself and can be changed over time in response to user actions or events. Props make components reusable, whereas state is used to create interactive and dynamic behavior within a component. Simply put: props are external and read-only, while state is internal and mutable.

### ****4. Explain ReactDOM.render()****

The ReactDOM.render() function is the method that tells React to display your application or component on the web page. It takes two arguments: the React element to render, and the DOM node where it should be mounted. For example, ReactDOM.render(<App />, document.getElementById('root')) tells React to render the App component into the HTML element with the id root. This is the entry point for most React applications, connecting your virtual React components with the actual browser DOM.