**• Define Props**

**Ans.** Props are read-only arguments passed into React components to transfer data from a parent component to a child component. For Example, a parent component passes a name prop to a child Welcome component:

**// Parent Component**

function App() {

return <Welcome name="Sara" />;

}

**// Child Component**

function Welcome(props) {

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

}

**• Explain Default Props**

**Ans.** Default props allow you to set default values for a component's props. This is useful when a parent component might not pass a specific prop. The component will use the default value if a prop is not provided. Example:

function Greeting({ name, punctuation }) {

return <h1>Hello, {name}{punctuation}</h1>;

}

**// Setting default values**

Greeting.defaultProps = {

name: "Guest",

punctuation: "!"

};

**// Renders "Hello, Guest!" because no props are passed.**

<Greeting />;

**// Renders "Hello, Jane!" because the name prop is provided, but punctuation uses the default.**

<Greeting name="Jane" />;

**• Identify the differences between States and Props**

**Ans.**

|  |  |
| --- | --- |
| **Props** | **States** |
| Immutable | Mutable |
| Passed down from a parent component | Initialized and managed within the component |
| Used to configure a component from the outside | Used to manage a component’s internal data |
| A component can’t change its own props | A component can update its own state |

**• Explain reactDOM.render()**

**Ans.** ReactDOM.render() is a method that was traditionally used to mount a React component tree into a specified container node in the DOM. It served as the entry point for a React application.

