**React - Components, State, Props**

**1- React Components:**

**Ans :-** In React, a component is a reusable, self-contained piece of UI that can be composed to build complex user interfaces. Components can be thought of as building blocks that encapsulate their own logic and rendering. Components can be either class components or functional components.

**Component type is two**

**Class Components: These are ES6 classes that extend React.Component and have a render() method to define the UI. They offer more features like state and lifecycle methods.**

class Welcome extends React.Component {

render() {

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

}

}

**Functional Components: These are JavaScript functions that return JSX (JavaScript XML) to define the UI. They are simpler and easier to test and understand.**

Syntext:-

function Welcome(props) {

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

}

jsx

1. **Props**

Ans :- Props (Properties): Props are used to pass data from a parent component to a child component. They are immutable, meaning they cannot be modified by the child component. Props are essential for making components reusable and customizable.

jsx  
Example:-

function Greeting(props) {

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

}

1. **State**

Ans:- State represents the internal data of a component. It is mutable and can be updated using the **setState()** method. State allows components to manage and reflect changes in their data over time, resulting in dynamic user interfaces.

Example:-

class Counter extends React.Component {

constructor(props) {

super(props);

this.state = { count: 0 };

}

increment() {

this.setState({ count: this.state.count + 1 });

}

render() {

return (

<div>

<p>Count: {this.state.count}</p>

<button onClick={() => this.increment()}>Increment</button>

</div>

);

}

}