**ReactJS-HOL 2**

**Questions & Answers –**

* **Explain React components**

**Ans -** React components are the building blocks of a React application. They are reusable, self-contained pieces of code that define how a part of the user interface should appear and behave. Each component can manage its own state and can be composed with other components to build complex UIs.

* **Identify the differences between components and JavaScript functions**

**Ans –**

| **Aspect** | **React Components** | **JavaScript Functions** |
| --- | --- | --- |
| **Purpose** | Used to create UI elements in React | Used to perform logic or return values |
| **JSX Usage** | Can return JSX (HTML-like syntax) | Returns values, usually not JSX |
| **Lifecycle Methods** | Can include lifecycle methods (in class comps) | No lifecycle methods |
| **React Features** | Can use props, state, hooks | Cannot use React features |
| **Integration with React DOM** | Rendered using ReactDOM.render() | Cannot be directly rendered in React DOM |

* **Identify the types of components**

**Ans -** There are mainly two types of components in React:

1. Class Components – Created using ES6 classes, can use lifecycle methods and state.
2. Function Components – Created using JavaScript functions, and can use hooks for state and lifecycle features.

* **Explain class component**

**Ans -** A class component is a React component defined using a JavaScript ES6 class that extends React.Component. It can maintain internal state, access lifecycle methods like componentDidMount(), and must define a render() method to return JSX.

**Example –**

class Welcome extends React.Component {

constructor(props) {

super(props);

this.state = { message: "Hello" };

}

render() {

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

}

}

* **Explain function component**

**Ans -** A function component is a simpler way to write components using JavaScript functions. With the introduction of hooks, function components can now handle state and side effects.

**Example –**

function Welcome(props) {

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

}

* **Define component constructor**

**Ans -** The constructor in a class component is a special method used to initialize state and bind methods. It is called when a component is created.

**Example –**

constructor(props) {

super(props);

this.state = { count: 0 };

}

* super(props) must be called to access this.props.
* It's optional unless you're initializing state or binding methods.
* **Define render() function**

**Ans -** The render() function is a required method in class components. It returns the JSX that should be displayed on the screen.

**Example –**

render() {

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

}

* Called automatically by React when rendering or updating the component.
* Can use this.props and this.state inside it.

**Hands-On – (Code)**

*App.js –*

import './App.css';

import About from './components/About';

import Contact from './components/Contact';

import Home from './components/Home';

function App() {

  return (

    <div className="container">

      <Home />

      <About />

      <Contact />

    </div>

  );

}

export default App;

*Home.js –*

import {Component} from 'react';

class Home extends Component{

    render(){

        return(

        <div>

        <h3>Welcome to the Home Page of Student Management Portal</h3>

        </div>

        );

    }

}

export default Home;

*About.js –*

import {Component} from 'react';

class About extends Component{

    render(){

        return(

        <div>

        <h3>Welcome to the About Page of Student Management Portal</h3>

        </div>

        );

    }

}

export default About;

*Contact.js –*

import {Component} from 'react';

class Contact extends Component{

    render(){

        return(

        <div>

        <h3>Welcome to the Contact Page of Student Management Portal</h3>

        </div>

        );

    }

}

export default Contact;





