**ReactJS-HOL 10**

**Question & Answers –**

1. **Define JSX**

**Ans -** JSX (JavaScript XML) is a syntax extension for JavaScript used in React. It allows you to write HTML-like code inside JavaScript. JSX makes it easier to create React elements visually and readably.

1. **Explain about ECMA Script**

**Ans -** ECMAScript (ES) is a standard for scripting languages like JavaScript. It defines the language's syntax, features, and rules. Modern JavaScript versions (like ES6, ES7, etc.) introduced features such as let, const, arrow functions, classes, modules, promises, and destructuring to improve coding efficiency and readability.

1. **Explain React.createElement()**

**Ans -** React.createElement() is a core method used by React to create virtual DOM elements. It's the JavaScript alternative to writing JSX.

**Example –** React.createElement('h1', {className: 'title'}, 'Hello React');

**This is equivalent to:** <h1 className="title">Hello React</h1>

1. **Explain how to create React nodes with JSX**

**Ans -** React nodes are typically created using JSX. You can define them as variables or return them from components.

**Example -** const element = <p>This is a React node</p>;

JSX gets compiled to React.createElement() calls behind the scenes.

1. **Define how to render JSX to DOM**

**Ans -** To render JSX into the DOM, use ReactDOM.render() in React 17 or createRoot in React 18+. **Example (React 18+):**

import React from 'react';

import ReactDOM from 'react-dom/client';

const element = <h1>Hello, React!</h1>;

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(element);

1. **Explain how to use JavaScript expressions in JSX**

**Ans -** In JSX, JavaScript expressions can be used by wrapping them in curly braces {}.

**Example –**

const name = 'Zoro';

const greeting = <h1>Hello, {name}!</h1>;

1. **Explain how to use inline CSS in JSX**

**Ans -** In JSX, inline styles are written as objects using camelCase properties, not hyphenated strings.

**Example -** <h2 style={{color: 'green', backgroundColor: 'lightgray'}}>Hello</h2>

**Hands-On – (Code)**

*OfficeSpace.js –*

const OfficeSpace = () => {

    const singleOffice = {

        name: 'DBS',

        rent: 50000,

        address: 'Chennai',

        image: '/images/chennai.jpg'

    };

    const officeList = [

    {

      name: 'DBS',

      rent: 50000,

      address: 'Chennai',

      image: '/images/chennai.jpg'

    },

    {

      name: 'Skyline View',

      rent: 70000,

      address: 'Mumbai',

      image: '/images/mumbai.jpg'

    },

    {

      name: 'Budget Office',

      rent: 45000,

      address: 'Hyderabad',

      image: '/images/hyderabad.jpg'

    }

  ];

  const getRentStyle = (rent) => {

    return{

    color: rent < 60000 ? 'red' : 'green',

    fontWeight: 'bold'

    }

  };

  return(

    <div style={{textAlign:'center', marginTop:'20px'}}>

        <h1>Office Space, at Affordable Range</h1>

        <img src={singleOffice.image} alt="Office Space" width="300" height="200"/>

        <h2>Name: {singleOffice.name}</h2>

        <p style={getRentStyle(singleOffice.rent)}>Rent: Rs. {singleOffice.rent}</p>

        <p><strong>Address: </strong>{singleOffice.address}</p>

        <hr style={{ margin: '40px 0' }} />

        <h2>More Office Listings</h2>

        {officeList.map((office, index) => (

            <div key={index} style={{ marginBottom: '30px' }}>

            <img src={office.image} alt={office.name} width="250" height="150" />

             <h3>Name: {office.name}</h3>

            <p style={getRentStyle(office.rent)}>Rent: Rs. {office.rent}</p>

            <p><strong>Address:</strong> {office.address}</p>

        </div>

        ))}

    </div>

  );

}

export default OfficeSpace;

*App.js –*

import './App.css';

import OfficeSpace from './OfficeSpace';

function App() {

  return (

    <OfficeSpace />

  );

}

export default App;

**Output –**





