**WEEK – 7(HandsOn)**

**React**

**10. ReactJS-HOL**

**Objectives**

**1. Define JSX**

JSX (JavaScript XML) is a syntax extension for JavaScript used with React.  
It allows you to write HTML-like code inside JavaScript, which gets transformed into React elements.

**Example:**

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

JSX makes your UI code more readable and concise.

**2. Explain ECMA Script**

**ECMAScript (ES)** is the **standardized version of JavaScript**.

* **ES6 (ECMAScript 2015)** introduced major features like:
  + let and const
  + Arrow functions
  + Classes
  + Template literals
  + Destructuring
  + Modules (import/export)
  + Promises

React is built using ES6+ syntax for cleaner and modern code.

**3. Explain React.createElement()**

Before JSX, React elements were created using the React.createElement() function.

**Syntax:**

React.createElement(type, props, children)

**Example:**

const element = React.createElement('h1', { className: 'title' }, 'Hello React');

**This is equivalent to JSX:**

const element = <h1 className="title">Hello React</h1>;

**4. How to Create React Nodes with JSX**

In JSX, a **React node** can be created just like writing HTML inside JS:

**Example:**

const heading = <h1>Welcome to Office Space</h1>;

const image = <img src="office.jpg" alt="Office" />;

**5. How to Render JSX to DOM**

In React, you render JSX into the DOM using ReactDOM.render() (in older versions) or using the root.render() method in React 18+.

**Example (index.js):**

import React from 'react';

import ReactDOM from 'react-dom/client';

import App from './App';

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

root.render(<App />);

JSX in <App /> gets rendered inside the div with id="root" in public/index.html.

**6. Using JavaScript Expressions in JSX**

You can embed any **valid JavaScript expression** inside JSX using {}.

**Example:**

const user = 'Priya';

const rent = 55000;

return (

<div>

<h1>Welcome, {user}</h1>

<p>Rent: Rs. {rent}</p>

<p>Status: {rent > 60000 ? "Expensive" : "Affordable"}</p>

</div>

);

**7. Using Inline CSS in JSX**

Use the style prop with a JavaScript object (camelCase style names):

**Example:**

<p style={{ color: 'red', fontWeight: 'bold' }}>Rent: Rs. 50000</p>

**CODE:**

**Src/App.js:**

import React from "react";

function App() {

  return (

    <div style={{ marginLeft: "80px", marginTop: "60px" }}>

      <h1>

        Office Space , at Affordable Range

      </h1>

      <img

        src="https://images.unsplash.com/photo-1573164574572-cb89e39749b4?auto=format&fit=crop&w=600&q=60"

        alt="office"

        style={{ width: "250px", height: "200px", objectFit: "cover", display: "block", marginTop: "25px" }}

      />

      <div style={{ marginTop: "30px" }}>

        <h2>

          Name: <span style={{ fontWeight: "bold" }}>DBS</span>

        </h2>

        <p style={{ color: "red", fontWeight: "bold", fontSize: "20px", margin: '0' }}>

          Rent: Rs. 50000

        </p>

        <p style={{ fontWeight: "bold" }}>

          Address: <span style={{ color: "black" }}>Chennai</span>

        </p>

      </div>

    </div>

  );

}

export default App;

**src/App.css:**

/\* src/App.css \*/

.App {

  text-align: center;

  font-family: Arial, sans-serif;

  padding: 20px;

}

h1 {

  color: navy;

}

img {

  border-radius: 8px;

  margin-bottom: 20px;

}

**OUTPUT:**

