**Office Space Rental App**

**REACT**

Vaishnavi

29/7/2025

**JSX (JavaScript XML)** is a syntax extension for JavaScript used in React to describe what the UI should look like. It allows developers to write HTML-like code inside JavaScript, making the code more readable and declarative. JSX is not mandatory in React, but it is widely used because it simplifies the process of creating UI components.

This lab demonstrates the use of **JSX in React** to:

Create elements and attributes.

Render them to the DOM.

Use **JavaScript expressions inside JSX**.

Apply **conditional inline CSS styling** based on business logic.

The project involves creating an application called **“Office Space Rental App”**, which displays office spaces, their details (Name, Rent, Address), and dynamically styles the rent amount in red or green based on the rent value.

**Objective:**

JSX stands for JavaScript XML.It allows writing HTML-like syntax inside JavaScript code, especially in React.JSX makes it easier to create and visualize UI components. It is not plain HTML, but it gets converted into JavaScript using a compiler like Babel.

ECMAScript is the official standard for scripting languages like JavaScript.  
It defines how the language should behave and what features it supports.  
React uses features from ECMAScript, especially ES6 and later, such as let, const, arrow functions, classes, promises, and modules.

The React.createElement() method is used by React to create virtual DOM elements.It takes three arguments: the type of element, its properties (if any), and its content.For example, React.createElement('h1', null, 'Hello') creates an element similar to <h1>Hello</h1>.When we write JSX, it is internally converted into React.createElement() calls.

React nodes can be created using JSX by writing code that looks like HTML inside JavaScript files. For example:const element = <h2>Welcome</h2>. To render JSX into the browser, we use a method provided by React. In React version 18 and above, we use ReactDOM.createRoot() to connect JSX with a real HTML element in the DOM.

JavaScript expressions can be used inside JSX by wrapping them in curly braces {}.This allows dynamic values to be displayed.

Inline CSS in JSX is written as a JavaScript object. Property names use camelCase instead of normal CSS syntax.

**Implementation:**

**Step 1: Create a React App**

****

**Step 2: Use the Component in App.js**

import React from 'react';

import './App.css';

function App() {

// Heading using JSX

const heading = <h1 style={{ textAlign: 'center', color: '#2c3e50' }}>Office Space Rental</h1>;

// Office object

const office = {

name: "Tech Park",

rent: 55000,

address: "MG Road, Bangalore",

image: "https://via.placeholder.com/300x200"

};

// List of offices

const officeList = [

{ id: 1, name: "Tech Park", rent: 55000, address: "MG Road, Bangalore", image: "https://via.placeholder.com/300x200" },

{ id: 2, name: "Work Hub", rent: 65000, address: "Whitefield, Bangalore", image: "https://via.placeholder.com/300x200" },

{ id: 3, name: "Sky Tower", rent: 45000, address: "Indiranagar, Bangalore", image: "https://via.placeholder.com/300x200" }

];

return (

<div className="App">

{heading}

<h2 style={{ marginTop: '20px' }}>Featured Office:</h2>

<div style={{ border: "1px solid #ccc", padding: "10px", width: "320px", margin: "10px auto" }}>

<img src={office.image} alt="Office" width="300" />

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

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

<p style={{ color: office.rent > 60000 ? 'green' : 'red' }}>Rent: ₹{office.rent}</p>

</div>

<h2 style={{ marginTop: '30px' }}>Available Office Spaces:</h2>

<div style={{ display: "flex", justifyContent: "center", gap: "15px", flexWrap: "wrap" }}>

{officeList.map((item) => (

<div key={item.id} style={{ border: "1px solid #ccc", padding: "10px", width: "300px", textAlign: "center" }}>

<img src={item.image} alt={item.name} width="280" />

<h3>{item.name}</h3>

<p>{item.address}</p>

<p style={{ color: item.rent > 60000 ? 'green' : 'red' }}>Rent: ₹{item.rent}</p>

</div>

))}

</div>

</div>

);

}

export default App;

**App.css**

.App {

  text-align: center;

  padding: 20px;

  font-family: Arial, sans-serif;

}

.heading {

  font-size: 32px;

  margin-bottom: 20px;

}

.office-container {

  display: flex;

  justify-content: space-around;

  flex-wrap: wrap;

  gap: 20px;

}

.office-card {

  border: 1px solid #ddd;

  border-radius: 10px;

  padding: 15px;

  width: 300px;

  box-shadow: 2px 2px 10px rgba(0,0,0,0.1);

}

.office-card img {

  width: 100%;

  border-radius: 8px;}

.low-rent {

  color: red;

  font-weight: bold;

}

.high-rent {

  color: green;

  font-weight: bold;

}

**Step 4: Run the App**

****

**Output:**

Rent in **green** if ≥ ₹60,000, **red** if below



