## **Objectives**

* Define JSX
* Explain about ECMA Script
* Explain React.createElement()
* Explain how to create React nodes with JSX
* Define how to render JSX to DOM
* Explain how to use JavaScript expressions in JSX
* Explain how to use inline CSS in JSX

In this hands-on lab, you will learn how to:

* Use JSX syntax in React applications
* Use inline CSS in JSX

## **Prerequisites**

The following is required to complete this hands-on lab:

* Node.js
* NPM
* Visual Studio Code

## **Notes**

Estimated time to complete this lab: **60 minutes.**

Create a React Application named “officespacerentalapp” which uses React JSX to create elements, attributes and renders DOM to display the page.

Create an element to display the heading of the page.

Attribute to display the image of the office space

Create an object of office to display the details like Name, Rent and Address.

Create a list of Object and loop through the office space item to display more data.

To apply Css, Display the color of the Rent in Red if it’s below 60000 and in Green if it’s above 60000.

**officespacerentalapp/src/App.js**

import React from "react";

// Featured single office object (demonstrates object usage)

const featuredOffice = {

id: 0,

name: "Regus Business Centre",

rent: 55000,

address: "MG Road, Bangalore",

// sample image from unsplash

image:

"https://images.unsplash.com/photo-1542314831-068cd1dbfeeb?auto=format&fit=crop&w=800&q=60",

};

// Array of office objects (demonstrates list/map)

const offices = [

featuredOffice,

{

id: 1,

name: "WeWork Galaxy",

rent: 75000,

address: "Residency Road, Bangalore",

image:

"https://images.unsplash.com/photo-1560448204-e02f11c3d0e2?auto=format&fit=crop&w=800&q=60",

},

{

id: 2,

name: "91Springboard Koramangala",

rent: 60000,

address: "Koramangala, Bangalore",

image:

"https://images.unsplash.com/photo-1524758631624-e2822e304c36?auto=format&fit=crop&w=800&q=60",

},

{

id: 3,

name: "Local Hub Office",

rent: 42000,

address: "Indiranagar, Bangalore",

image:

"https://images.unsplash.com/photo-1505672678657-cc7037095e2c?auto=format&fit=crop&w=800&q=60",

},

];

// Small utility: produces an inline style object for rent color

const rentStyle = (rent) => ({ color: rent < 60000 ? "red" : "green" });

// OfficeCard component (kept inside same file for simplicity)

function OfficeCard({ office }) {

return (

<div

style={{

border: "1px solid #ddd",

borderRadius: 8,

padding: 12,

marginBottom: 12,

display: "flex",

gap: 12,

alignItems: "center",

}}

>

<img

src={office.image}

alt={office.name}

style={{ width: 140, height: 90, objectFit: "cover", borderRadius: 6 }}

/>

<div>

{/\* JSX expression: embed JS values inside JSX \*/}

<h3 style={{ margin: 0 }}>{office.name}</h3>

<p style={{ margin: "6px 0 0 0" }}>{office.address}</p>

{/\* Inline CSS using a JS object and conditional color logic \*/}

<p style={{ margin: "6px 0 0 0", fontWeight: 600 }}>

Rent: <span style={rentStyle(office.rent)}>{`₹${office.rent.toLocaleString()}`}</span>

</p>

</div>

</div>

);

}

export default function App() {

return (

<div style={{ maxWidth: 900, margin: "24px auto", padding: 16, fontFamily: "Segoe UI, Roboto, Arial" }}>

<header style={{ textAlign: "center", marginBottom: 20 }}>

<h1 style={{ margin: 0 }}>Office Space Rental</h1>

<p style={{ marginTop: 6, color: "#555" }}>Using JSX, inline CSS and JS expressions</p>

</header>

{/\* Single object example: featured office \*/}

<section style={{ marginBottom: 18 }}>

<h2 style={{ marginBottom: 8 }}>Featured Office</h2>

<OfficeCard office={featuredOffice} />

</section>

<hr />

{/\* List example: map through the offices array and render OfficeCard for each \*/}

<section style={{ marginTop: 18 }}>

<h2>Available Offices</h2>

{/\* map() demonstrates list rendering with JSX \*/}

{offices.map((office) => (

<OfficeCard key={office.id} office={office} />

))}

</section>

<footer style={{ marginTop: 20, color: "#666", fontSize: 13 }}>

<p>Note: Rent values below 60000 are shown in red, rents 60000 and above are green.</p>

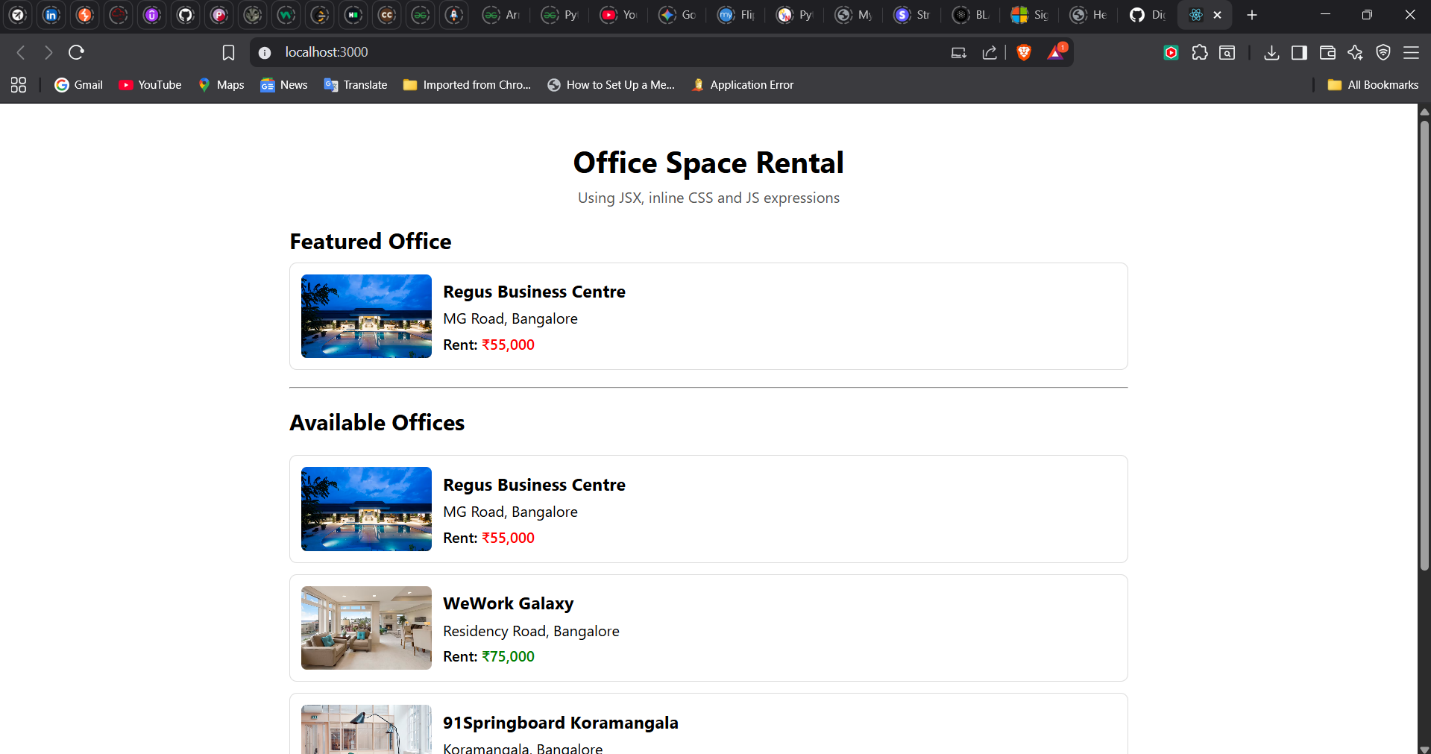
</footer>

</div>

);

}

Output:



**Hint:**



