**Objectives**

* List the features of ES6
* Explain JavaScript let
* Identify the differences between var and let
* Explain JavaScript const
* Explain ES6 class fundamentals
* Explain ES6 class inheritance
* Define ES6 arrow functions
* Identify set(), map()

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

* Use map() method of ES6
* Apply arrow functions of ES6
* Implement Destructuring features of ES6

**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 “cricketapp” with the following components:

1. ListofPlayers

* Declare an array with 11 players and store details of their names and scores using the map feature of ES6



* Filter the players with scores below 70 using arrow functions of ES6.



1. IndianPlayers
   1. Display the Odd Team Player and Even Team players using the Destructuring features of ES6



* 1. Declare two arrays T20players and RanjiTrophy players and merge the two arrays and display them using the Merge feature of ES6



Display these two components in the same home page using a simple if else in the flag variable.

**Output:**

When Flag=true



When Flag=false



**Hint:**



**File: ListofPlayers.js**

// ListofPlayers.js

import React from 'react';

const ListofPlayers = () => {

const players = [

{ name: 'Virat Kohli', score: 85 },

{ name: 'Rohit Sharma', score: 95 },

{ name: 'Hardik Pandya', score: 65 },

{ name: 'Ravindra Jadeja', score: 55 },

{ name: 'KL Rahul', score: 75 },

];

const filteredPlayers = players

.map(player => ({

...player,

status: player.score >= 70 ? 'Selected' : 'Not Selected',

}));

return (

<div>

<h2>List of Players</h2>

<ul>

{filteredPlayers.map((player, index) => (

<li key={index}>

{player.name} - Score: {player.score} - Status: {player.status}

</li>

))}

</ul>

</div>

);

};

export default ListofPlayers;

**File: IndianPlayers.js**

// IndianPlayers.js

import React from 'react';

const IndianPlayers = () => {

const t20 = ['Virat', 'Rohit', 'Hardik'];

const ranji = ['Shubman', 'Sarfaraz', 'Shreyas'];

const allPlayers = [...t20, ...ranji];

const oddTeam = allPlayers.filter((\_, index) => index % 2 !== 0);

const evenTeam = allPlayers.filter((\_, index) => index % 2 === 0);

return (

<div>

<h2>Indian Players</h2>

<p><strong>All Players:</strong> {allPlayers.join(', ')}</p>

<p><strong>Even Team:</strong> {evenTeam.join(', ')}</p>

<p><strong>Odd Team:</strong> {oddTeam.join(', ')}</p>

</div>

);

};

export default IndianPlayers;

**File: App.js**

// App.js

import React, { useState } from 'react';

import ListofPlayers from './ListofPlayers';

import IndianPlayers from './IndianPlayers';

const App = () => {

const [showList, setShowList] = useState(true);

const toggleComponent = () => {

setShowList(!showList);

};

return (

<div>

<button onClick={toggleComponent}>

{showList ? 'Show Indian Players' : 'Show List of Players'}

</button>

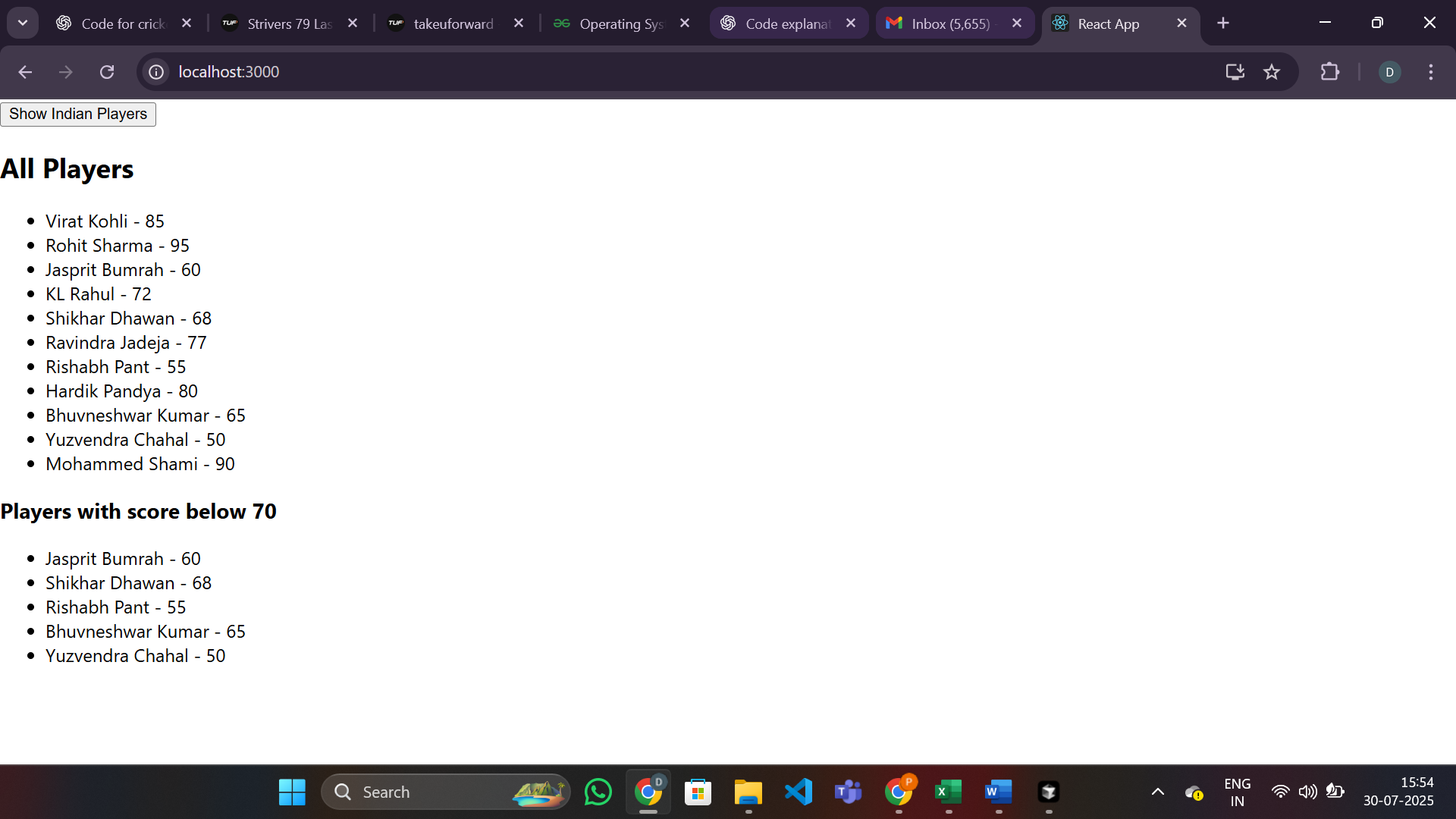
{showList ? <ListofPlayers /> : <IndianPlayers />}

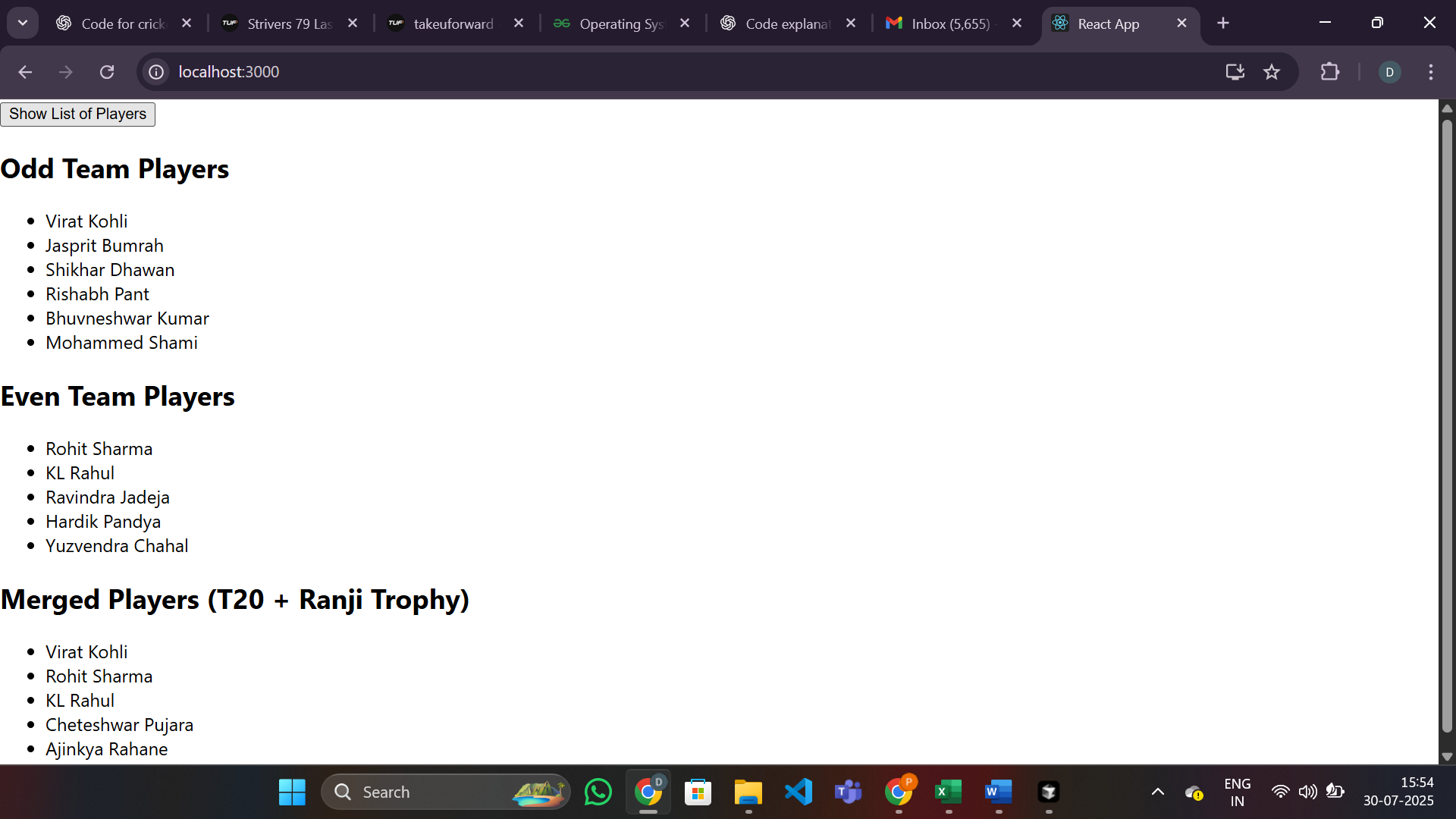
</div>

);

};

export default App;





**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.

Output:



**Hint:**





App.js

import React from "react";

*// Office image URL (replace with your own if needed)*

const officeImage = "https://images.unsplash.com/photo-1506744038136-46273834b3fb?auto=format&fit=crop&w=600&q=80";

*// Single office object*

const office = {

  name: "Downtown Workspace",

  rent: 55000,

  address: "123 Main St, City Center"

};

*// List of office objects*

const officeList = [

  {

    name: "Downtown Workspace",

    rent: 55000,

    address: "123 Main St, City Center"

  },

  {

    name: "Uptown Executive Suite",

    rent: 75000,

    address: "456 Uptown Ave, Business District"

  },

  {

    name: "Suburban Office Hub",

    rent: 40000,

    address: "789 Suburb Rd, Suburbia"

  }

];

*// Function to get rent color*

const **getRentColor** = (*rent*) => ({

  color: *rent* < 60000 ? "red" : "green",

  fontWeight: "bold"

});

function **App**() {

  return (

    <div *style*={{ fontFamily: "Arial, sans-serif", padding: 20 }}>

      {*/\* Heading using JSX \*/*}

      <h1>Office Space Rental App</h1>

      {*/\* Office image with JSX attribute \*/*}

      <img *src*={officeImage} *alt*="Office Space" *style*={{ width: 300, borderRadius: 8 }} />

      {*/\* Display single office object \*/*}

      <h2>Featured Office</h2>

      <div *style*={{ border: "1px solid #ccc", padding: 10, marginBottom: 20, borderRadius: 8 }}>

        <p><strong>Name:</strong> {office.name}</p>

        <p>

          <strong>Rent:</strong>

          <span *style*={getRentColor(office.rent)}> ₹{office.rent}</span>

        </p>

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

      </div>

      {*/\* Display list of offices \*/*}

      <h2>Available Offices</h2>

      <ul *style*={{ listStyle: "none", padding: 0 }}>

        {officeList.map((*item*, *idx*) => (

          <li *key*={idx} *style*={{ border: "1px solid #eee", marginBottom: 10, padding: 10, borderRadius: 8 }}>

            <p><strong>Name:</strong> {item.name}</p>

            <p>

              <strong>Rent:</strong>

              <span *style*={getRentColor(item.rent)}> ₹{item.rent}</span>

            </p>

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

          </li>

        ))}

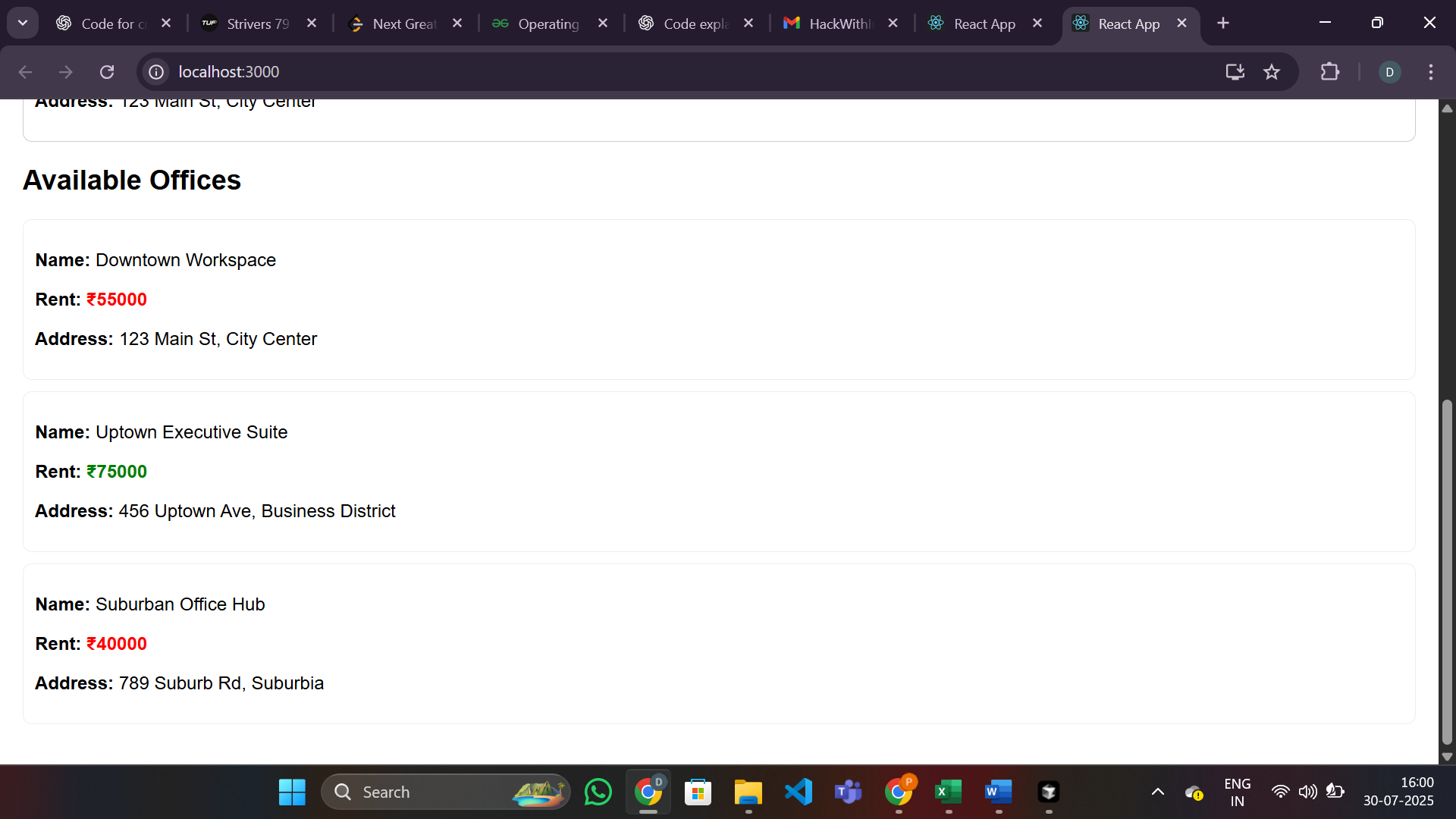
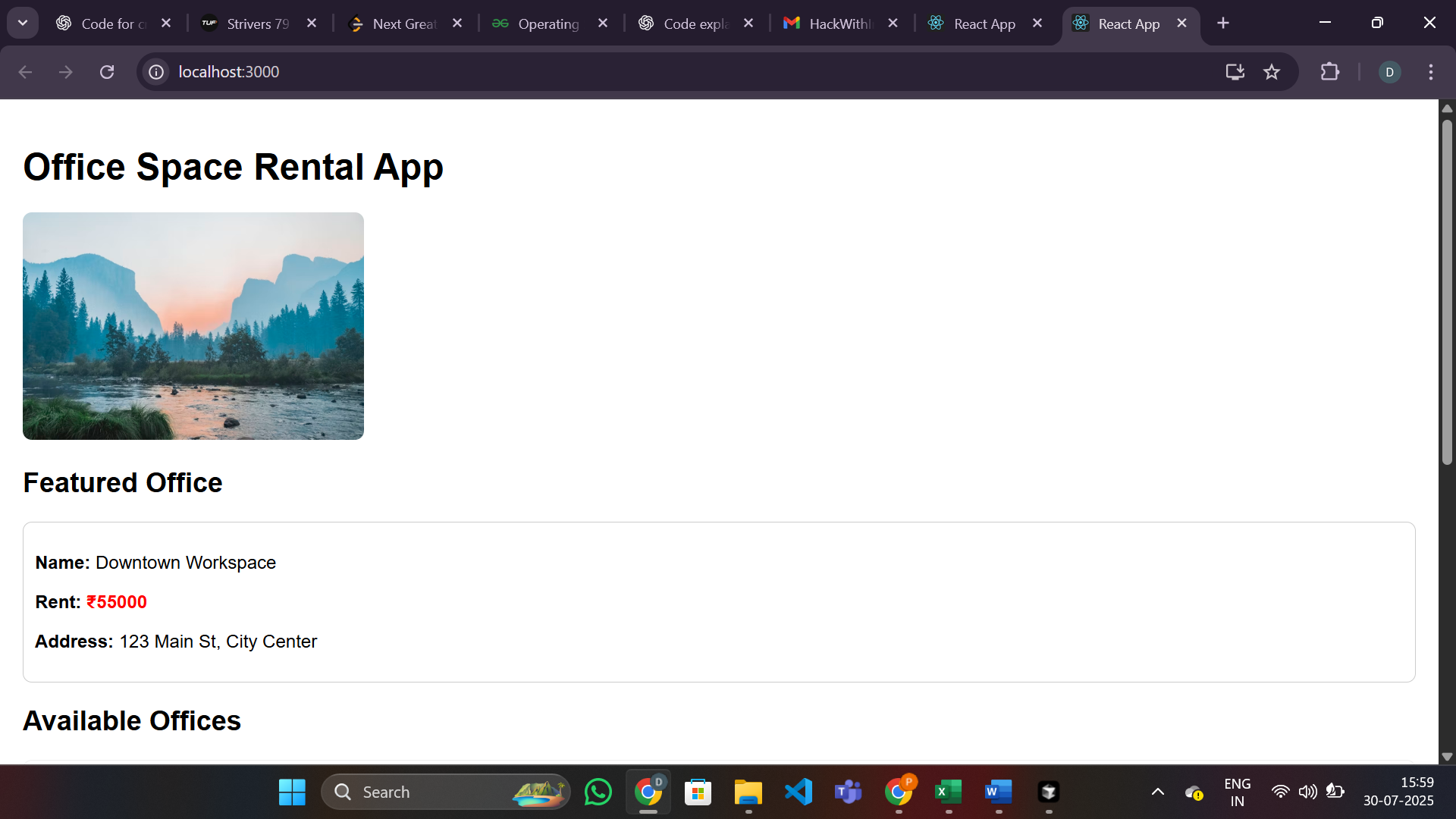
      </ul>

    </div>

  );

}

export default App;



**11. ReactJS-HOL.docx:**

**Objectives:**

* Explain React events
* Explain about event handlers
* Define Synthetic event
* Identify React event naming convention

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

* Implement Event handling in React applications
* Use this keyword
* Use synthetic events

**Prerequisites:**

* Node.js
* NPM
* Visual Studio Code

**Notes:**

* Estimated time to complete this lab: 90 minutes.
* Create a React Application **“eventexamplesapp”** to handle various form events.

**Functional Requirements:**

* “**Increment**” button → increase the counter
* “**Decrement**” button → decrease the counter
* The **Increase** button should invoke:
  1. Method to increment value
  2. Method to say Hello + static message
* A button “**Say Welcome**” → passes "welcome" as an argument
* A button with synthetic event **OnPress** → displays "I was clicked"
* A component **“CurrencyConvertor”**:
  1. Converts Indian Rupees to Euro on button click
  2. Handles the onClick event to call handleSubmit for conversion

CurrencyConvertor.js

import React, { Component } from "react";

class CurrencyConvertor extends Component {

  constructor(*props*) {

*super*(*props*);

    this.state = {

      rupees: "",

      euro: ""

    };

  }

**handleChange** = (*e*) => {

    this.setState({ rupees: *e*.target.value });

  };

**handleSubmit** = (*e*) => {

*e*.preventDefault(); *// Synthetic event*

    const euro = (parseFloat(this.state.rupees) / 90).toFixed(2); *// Example rate*

    this.setState({ euro });

  };

**render**() {

    return (

      <div>

        <h2>Currency Convertor (INR to Euro)</h2>

        <form *onSubmit*={this.handleSubmit}>

          <input

*type*="number"

*value*={this.state.rupees}

*onChange*={this.handleChange}

*placeholder*="Enter amount in INR"

          />

          <button *type*="submit">Convert</button>

        </form>

        {this.state.euro && (

          <div>

            <strong>Euro: €{this.state.euro}</strong>

          </div>

        )}

      </div>

    );

  }

}

export default CurrencyConvertor;

App.js

import React, { Component } from "react";

import CurrencyConvertor from "./CurrencyConvertor";

class App extends Component {

  constructor(*props*) {

*super*(props);

    this.state = {

      counter: 0,

      message: ""

    };

*// Bind methods if not using arrow functions*

    this.increment = this.increment.bind(this);

    this.sayHello = this.sayHello.bind(this);

  }

  increment() {

    this.setState({ counter: this.state.counter + 1 });

    this.sayHello();

  }

  decrement = () => {

    this.setState({ counter: this.state.counter - 1 });

  };

  sayHello() {

    this.setState({ message: "Hello! You clicked Increment." });

  }

  sayWelcome = (*msg*) => {

    this.setState({ message: `Welcome! Message: ${msg}` });

  };

  handleSyntheticEvent = (*e*) => {

*// e is a SyntheticEvent*

    this.setState({ message: "I was clicked (Synthetic Event)!" });

  };

  render() {

    return (

      <div *style*={{ padding: 20 }}>

        <h1>React Event Handling Examples</h1>

        <h2>Counter: {this.state.counter}</h2>

        <button *onClick*={this.increment}>Increment</button>

        <button *onClick*={this.decrement}>Decrement</button>

        <br /><br />

        <button *onClick*={() => this.sayWelcome("welcome")}>Say Welcome</button>

        <br /><br />

        <button *onClick*={this.handleSyntheticEvent}>Synthetic Event OnPress</button>

        <br /><br />

        <div *style*={{ color: "blue" }}>{this.state.message}</div>

        <hr />

        <CurrencyConvertor />

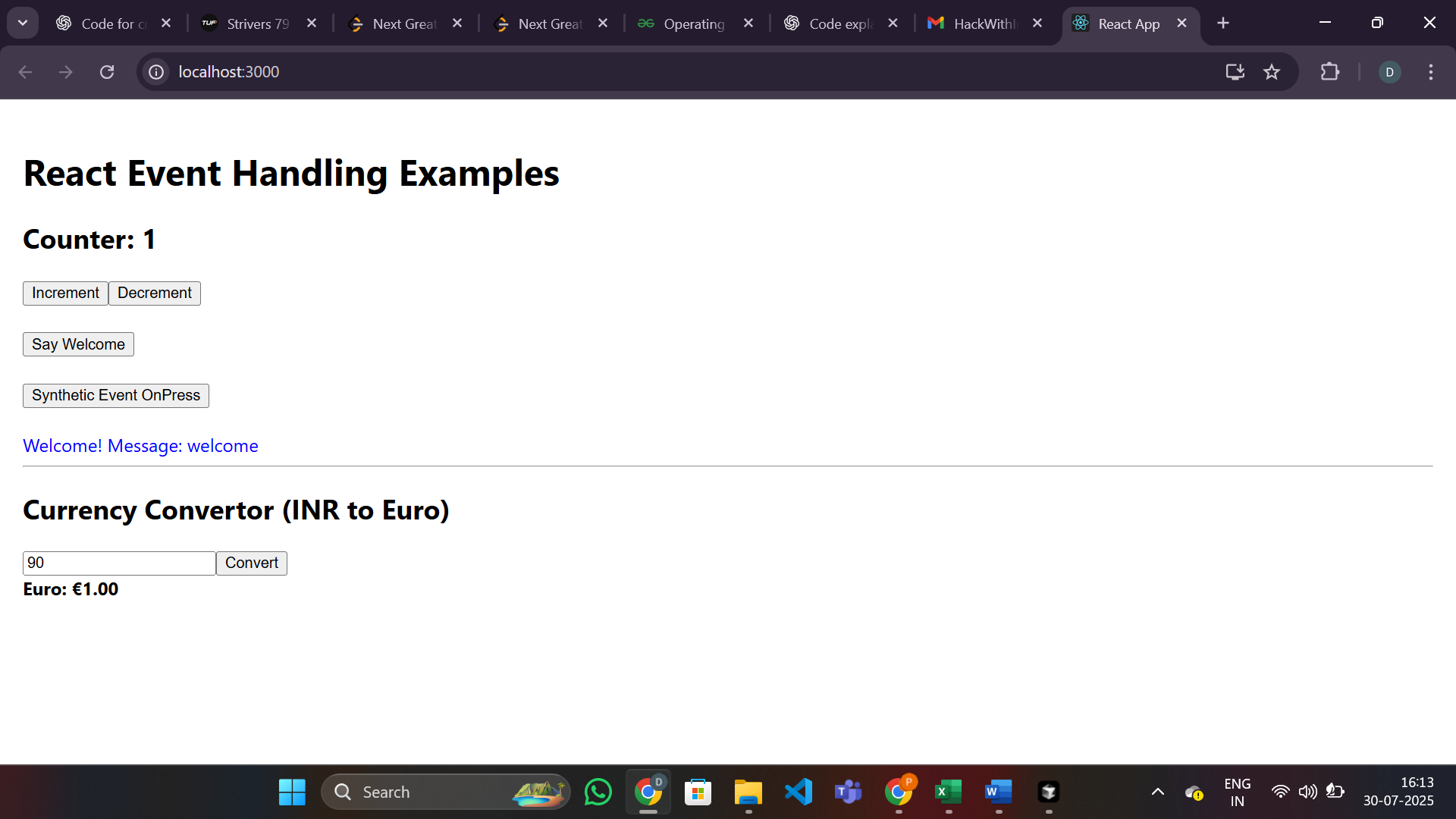
      </div>

    );

  }

}

export default App;



**12. ReactJS-HOL.docx:**

**Objectives:**

* Explain about conditional rendering in React
* Define element variables
* Explain how to prevent components from rendering

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

* Implement conditional rendering in React applications

**Prerequisites:**

* Node.js
* NPM
* Visual Studio Code

**Notes:**

* Estimated time to complete this lab: 60 minutes.
* Create a React Application named **“ticketbookingapp”** where:
  + Guest users can browse flight details.
  + Logged-in users can **book tickets**.
  + The **Login** and **Logout** buttons should toggle the pages.
  + Once logged in → **User Page** is displayed.
  + After logging out → **Guest Page** is shown.

App.js

import React, { useState } from "react";

*// Guest page component*

function **GuestPage**() {

  return (

    <div>

      <h2>Welcome, Guest!</h2>

      <p>Browse available flights below:</p>

      <ul>

        <li>Flight A: Mumbai → Delhi, 10:00 AM</li>

        <li>Flight B: Bangalore → Chennai, 12:00 PM</li>

        <li>Flight C: Kolkata → Hyderabad, 3:00 PM</li>

      </ul>

    </div>

  );

}

*// User page component*

function **UserPage**({ *onLogout* }) {

  const [booked, **setBooked**] = useState({ A: false, B: false, C: false });

  const **handleBook** = (*flight*) => {

    setBooked(*prev* => ({ ...*prev*, [*flight*]: true }));

  };

  return (

    <div>

      <h2>Welcome, User!</h2>

      <p>You can now book your tickets:</p>

      <div>

        <button *onClick*={() => handleBook('A')} *disabled*={booked.A}>Book Flight A</button>

        {booked.A && <span *style*={{ color: 'green', marginLeft: 8 }}>Booked</span>}

      </div>

      <div>

        <button *onClick*={() => handleBook('B')} *disabled*={booked.B}>Book Flight B</button>

        {booked.B && <span *style*={{ color: 'green', marginLeft: 8 }}>Booked</span>}

      </div>

      <div>

        <button *onClick*={() => handleBook('C')} *disabled*={booked.C}>Book Flight C</button>

        {booked.C && <span *style*={{ color: 'green', marginLeft: 8 }}>Booked</span>}

      </div>

      <br /><br />

      <button *onClick*={*onLogout*}>Logout</button>

    </div>

  );

}

function **App**() {

  const [isLoggedIn, setIsLoggedIn] = useState(false);

*// Element variable for conditional rendering*

  let pageContent;

  if (isLoggedIn) {

    pageContent = <UserPage *onLogout*={() => setIsLoggedIn(false)} />;

  } else {

    pageContent = <GuestPage />;

  }

  return (

    <div *style*={{ padding: 20 }}>

      <h1>Ticket Booking App</h1>

      {*/\* Login/Logout button \*/*}

      {!isLoggedIn ? (

        <button *onClick*={() => setIsLoggedIn(true)}>Login</button>

      ) : null}

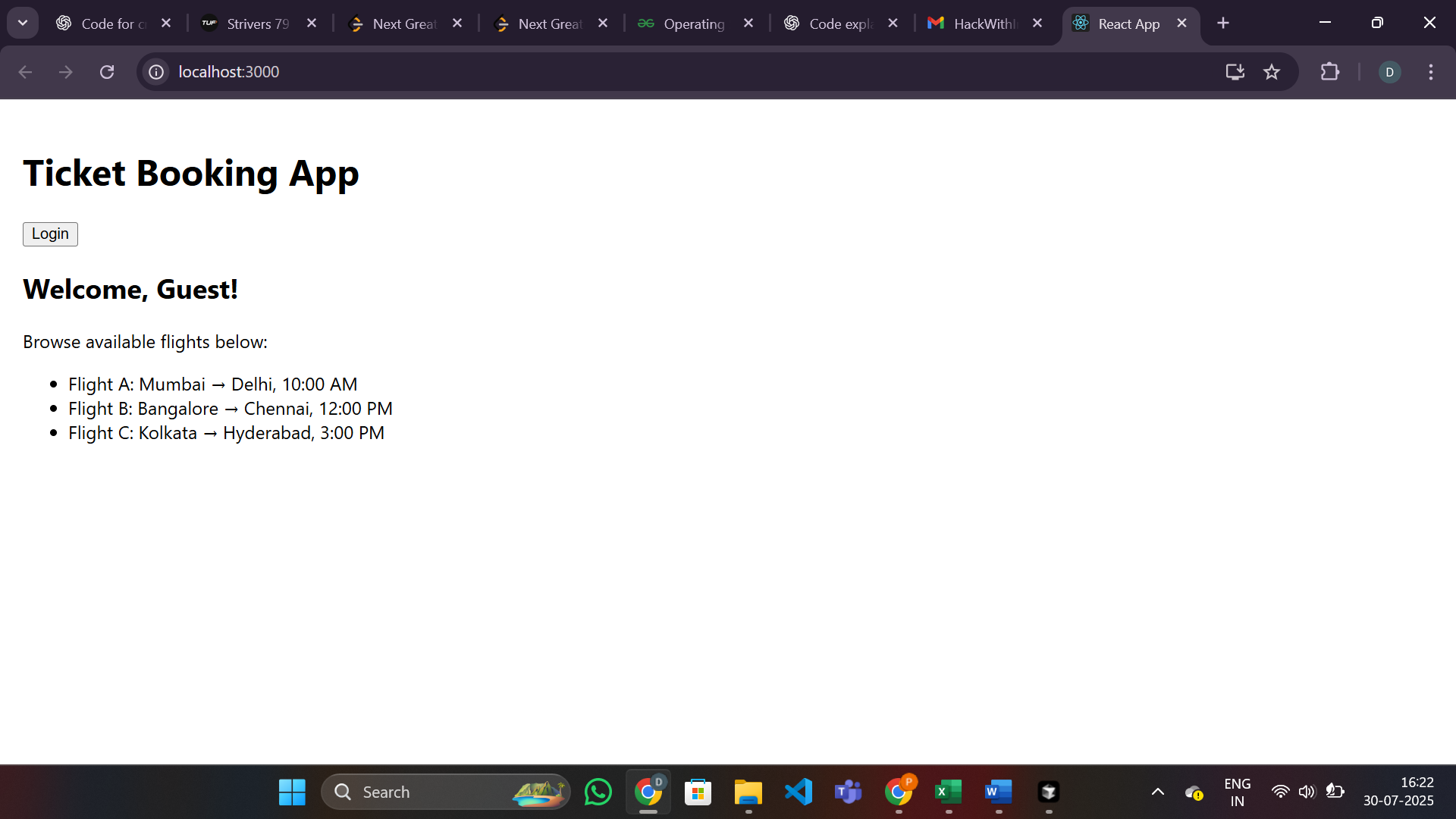
      {pageContent}

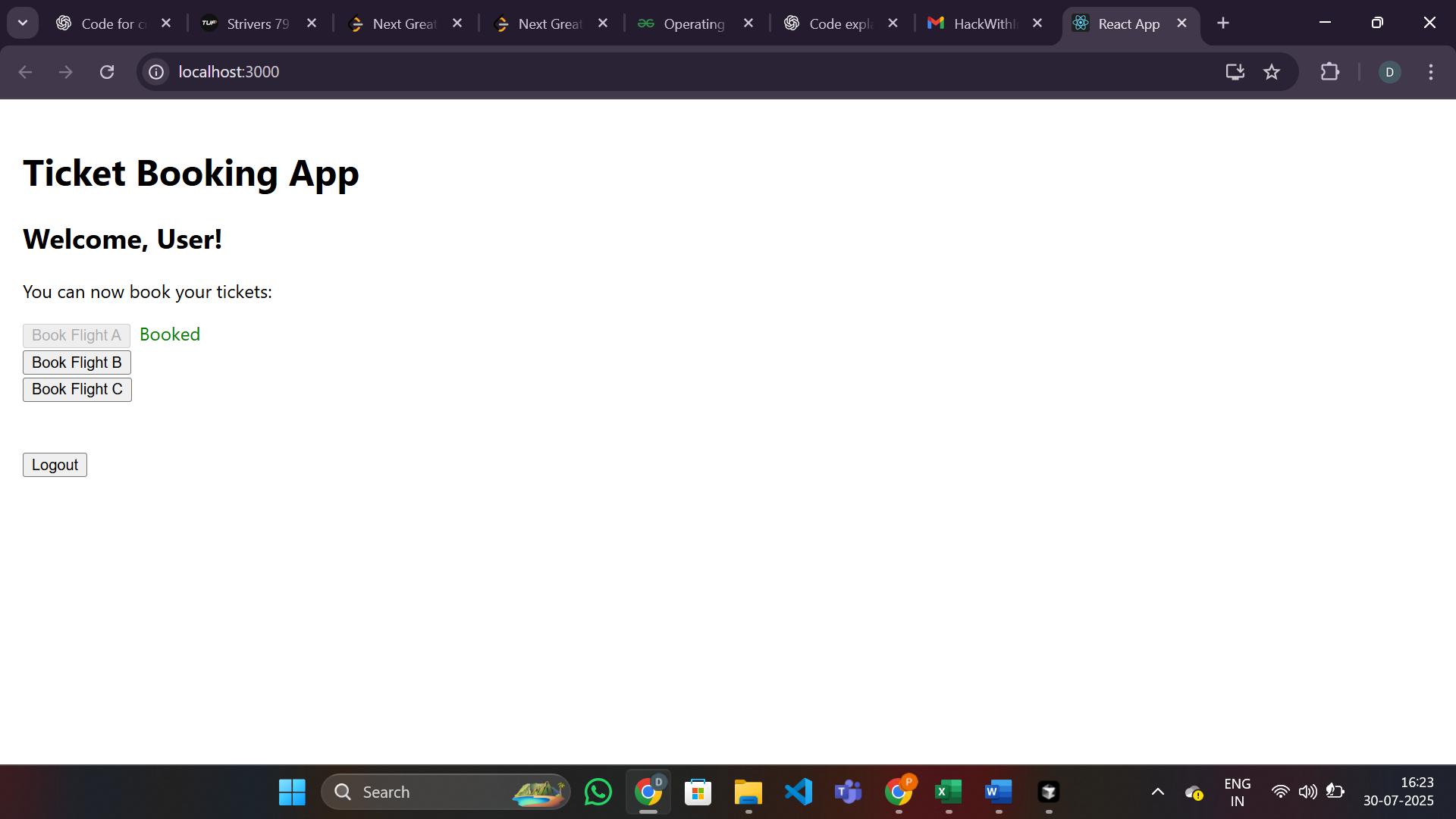
    </div>

  );

}

export default App;





**13. ReactJS-HOL.docx:**

**Objectives:**

* Explain various ways of conditional rendering
* Explain how to render multiple components
* Define list component
* Explain about keys in React applications
* Explain how to extract components with keys
* Explain React Map, map() function

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

* Implement conditional rendering in React applications

**Prerequisites:**

* Node.js
* NPM
* Visual Studio Code

**Notes:**

* Estimated time to complete this lab: 60 minutes.
* Create a React App named **“bloggerapp”** with 3 components:
  + Book Details
  + Blog Details
  + Course Details
* Implement this with as many ways possible of **Conditional Rendering**.

BookDetails.js

import React from "react";

const books = [

  { id: 1, title: "React in Action", author: "Mark Tielens Thomas" },

  { id: 2, title: "Learning React", author: "Alex Banks" },

  { id: 3, title: "Fullstack React", author: "Anthony Accomazzo" }

];

function **BookItem**({ *book* }) {

  return (

    <li *key*={*book*.id}>

      <strong>{*book*.title}</strong> by {*book*.author}

    </li>

  );

}

function **BookDetails**() {

  return (

    <div>

      <h2>Book Details</h2>

      <ul>

        {books.map(*book* => (

          <BookItem *key*={*book*.id} *book*={*book*} />

        ))}

      </ul>

    </div>

  );

}

export default BookDetails;

BlogDetails.js

import React from "react";

const blogs = [

  { id: 1, title: "React Patterns", author: "Michael Chan" },

  { id: 2, title: "State Management", author: "Dan Abramov" }

];

function **BlogDetails**() {

  return (

    <div>

      <h2>Blog Details</h2>

      <ul>

        {blogs.map(*blog* => (

          <li *key*={*blog*.id}>

            <strong>{*blog*.title}</strong> by {*blog*.author}

          </li>

        ))}

      </ul>

    </div>

  );

}

export default BlogDetails;

CourseDetails.js

import React from "react";

const courses = [

  { id: 1, name: "React Basics", duration: "4 weeks" },

  { id: 2, name: "Advanced React", duration: "6 weeks" }

];

function **CourseDetails**() {

  return (

    <div>

      <h2>Course Details</h2>

      <ul>

        {courses.map(*course* => (

          <li *key*={*course*.id}>

            <strong>{*course*.name}</strong> - {*course*.duration}

          </li>

        ))}

      </ul>

    </div>

  );

}

export default CourseDetails;

App.js

import React, { useState } from "react";

import BookDetails from "./BookDetails";

import BlogDetails from "./BlogDetails";

import CourseDetails from "./CourseDetails";

function **App**() {

  const [view, **setView**] = useState("books");

  const [showCourses, **setShowCourses**] = useState(true);

*// 1. If/else statement*

  let detailsComponent;

  if (view === "books") {

    detailsComponent = <BookDetails />;

  } else if (view === "blogs") {

    detailsComponent = <BlogDetails />;

  } else {

    detailsComponent = <CourseDetails />;

  }

  return (

    <div *style*={{ padding: 20 }}>

      <h1>Blogger App</h1>

      <button *onClick*={() => setView("books")}>Show Books</button>

      <button *onClick*={() => setView("blogs")}>Show Blogs</button>

      <button *onClick*={() => setView("courses")}>Show Courses</button>

      <br /><br />

      {*/\* 2. Element variable \*/*}

      {detailsComponent}

      {*/\* 3. Ternary operator \*/*}

      <div>

        <h3>Quick View (Ternary):</h3>

        {view === "books" ? <BookDetails /> : <BlogDetails />}

      </div>

      {*/\* 4. Logical && operator \*/*}

      <div>

        <h3>Show Courses (Logical &&):</h3>

        <button *onClick*={() => setShowCourses(!showCourses)}>

          {showCourses ? "Hide" : "Show"} Courses

        </button>

        {showCourses && <CourseDetails />}

      </div>

      {*/\* 5. Early return \*/*}

      {view === "none" && <div>No details to show.</div>}

    </div>

  );

}

export default App;

