**COGNIZANT - DIGITALNURTURE4.0**

**DEEPSKILLING JAVA FSE**

**WEEK 07: REACT**

**Exercise 1: ES6**

1. **List the features of ES6**  
   ES6 introduced features like let and const for better variable scoping, arrow functions for concise function syntax, classes and inheritance, template literals, destructuring, and new data structures like Map and Set.  
   It also added promises for asynchronous programming and modules for better code organization.
2. **Explain JavaScript let**  
   The let keyword allows you to declare block-scoped variables, meaning they are accessible only within the block where they are defined.  
   This helps avoid bugs caused by variables leaking outside loops or conditionals.
3. **Identify the differences between var and let**  
   var is function-scoped and can be redeclared, while let is block-scoped and cannot be redeclared in the same scope.  
   Using let makes code more predictable by limiting variable scope.
4. **Explain JavaScript const**  
   const declares a variable whose value cannot be reassigned after it is initialized.  
   However, if the value is an object or array, its contents can still be modified.
5. **Explain ES6 class fundamentals**  
   ES6 classes provide a clearer and structured way to create objects and define methods.  
   They use the class and constructor keywords to initialize properties and add functionality.
6. **Explain ES6 class inheritance**  
   ES6 allows classes to inherit from other classes using the extends keyword.  
   The super keyword inside the subclass constructor calls the parent class's constructor.
7. **Define ES6 arrow functions**  
   Arrow functions offer a shorter syntax for writing functions using =>.  
   They do not have their own this binding, making them useful in callbacks.
8. **Identify Set() and Map()**  
   Set is a collection of unique values that cannot have duplicates.  
   Map is a collection of key–value pairs where keys can be of any data type.

**// ListofPlayers.js**

import React from 'react';

const players = [

{ name: 'Sachin Tendulkar', score: 95 },

{ name: 'Virat Kohli', score: 87 },

{ name: 'Rohit Sharma', score: 90 },

{ name: 'MS Dhoni', score: 82 },

{ name: 'Yuvraj Singh', score: 66 },

{ name: 'Rahul Dravid', score: 45 },

{ name: 'Sourav Ganguly', score: 72 },

{ name: 'Jasprit Bumrah', score: 50 },

{ name: 'Ravindra Jadeja', score: 35 },

{ name: 'Shikhar Dhawan', score: 77 },

{ name: 'Harbhajan Singh', score: 68 }

];

const ListofPlayers = () => {

const belowSeventy = players.filter(player => player.score < 70);

return (

<div>

<h2>📋 All Players</h2>

<ul>

{players.map(player => (

<li key={player.name}>{player.name} – {player.score} runs</li>

))}

</ul>

<h2>⚠️ Players with Score Below 70</h2>

<ul>

{belowSeventy.map(player => (

<li key={player.name}>{player.name} – {player.score} runs</li>

))}

</ul>

</div>

);

};

export default ListofPlayers;

**// IndianPlayers.js**

import React from 'react';

const T20players = ['Rinku Singh', 'Ishan Kishan', 'Tilak Varma', 'Washington Sundar', 'Arshdeep Singh'];

const RanjiTrophyPlayers = ['Cheteshwar Pujara', 'Manoj Tiwary', 'Mayank Agarwal', 'Priyank Panchal', 'Devdutt Padikkal'];

const IndianPlayers = () => {

const allPlayers = [...T20players, ...RanjiTrophyPlayers];

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

const evenPlayers = allPlayers.filter((\_, idx) => idx % 2 === 0);

return (

<div>

<h2>🔁 Merged Indian Players</h2>

<ul>

{allPlayers.map(player => (

<li key={player}>{player}</li>

))}

</ul>

<h2>🟠 Odd Index Players</h2>

<ul>

{oddPlayers.map(player => (

<li key={player}>{player}</li>

))}

</ul>

<h2>🔵 Even Index Players</h2>

<ul>

{evenPlayers.map(player => (

<li key={player}>{player}</li>

))}

</ul>

</div>

);

};

export default IndianPlayers;

**//App.js**

import React from 'react';

import ListofPlayers from './ListofPlayers';

import IndianPlayers from './IndianPlayers';

const flag = true; // Toggle this to false to switch views

function App() {

return (

<div className="App">

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

</div>

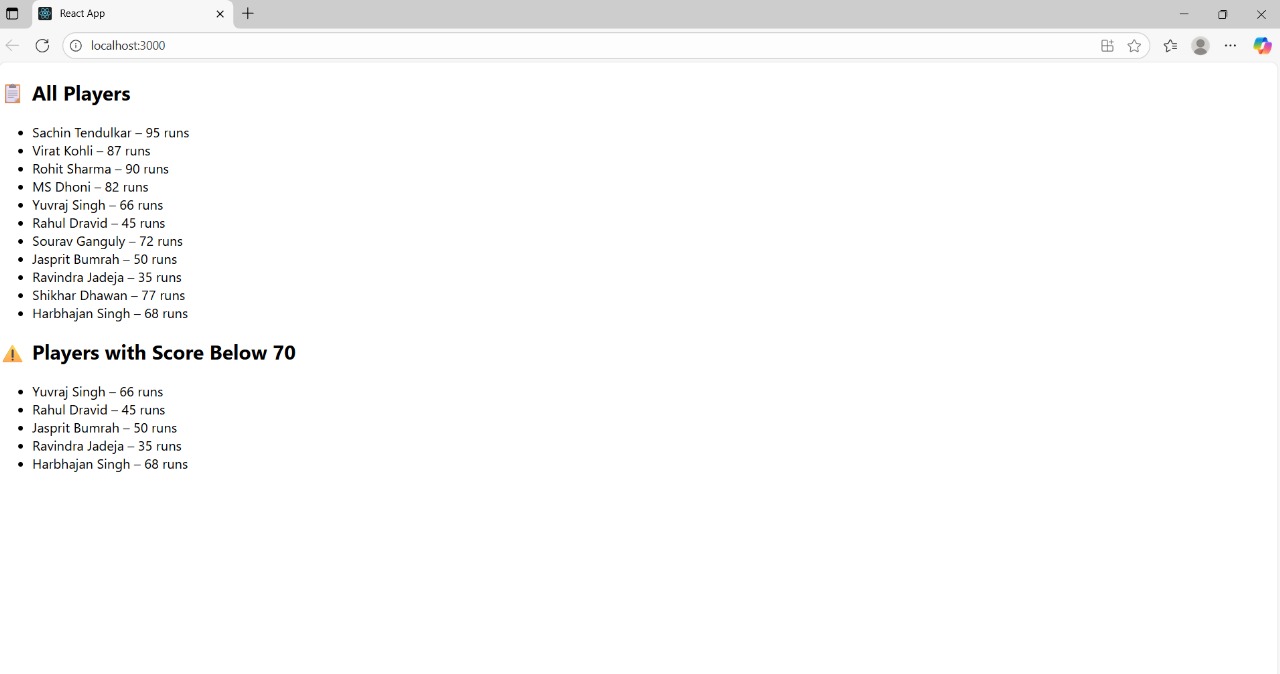
);

}

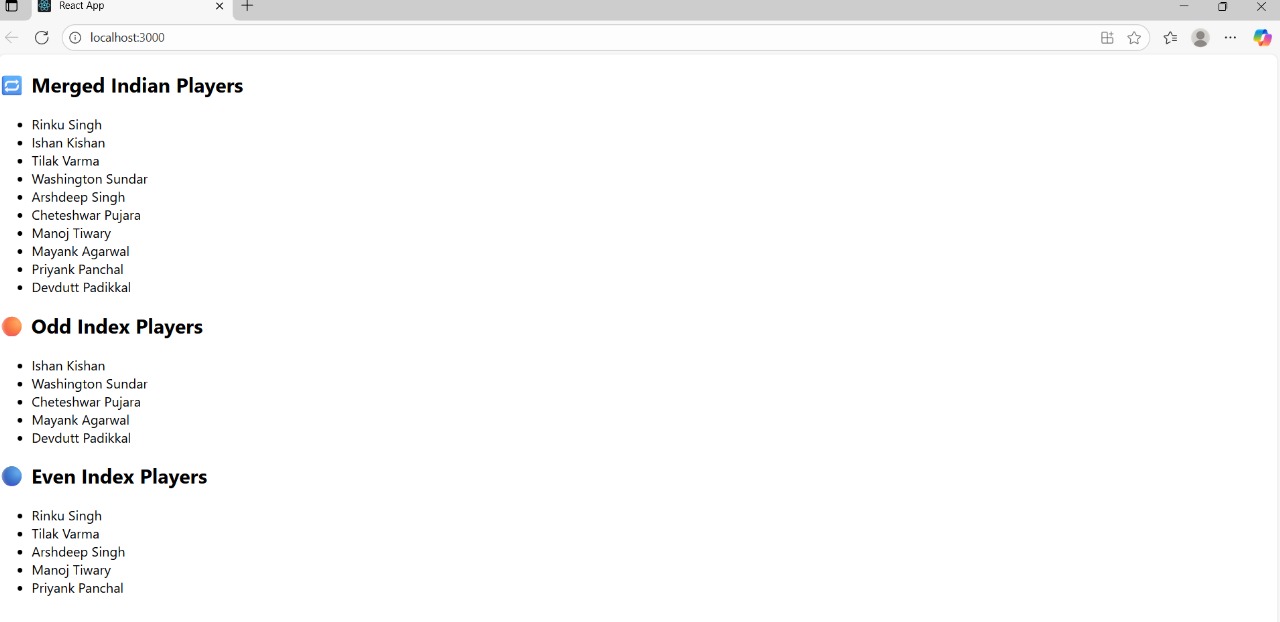
export default App;

**OUTPUT:**

* When Flag = true



* When Flag = false



**Exercise 2: JSX**

1. **Define JSX**  
   JSX stands for JavaScript XML and allows writing HTML-like syntax directly within JavaScript code.  
   It makes defining React UI components easier and more readable.
2. **Explain about ECMA Script**  
   ECMA Script is the standardized scripting language specification that JavaScript is based on.  
   It defines rules, syntax, and features that make JavaScript consistent across different environments.
3. **Explain React.createElement()**  
   React.createElement() is a method used to create React elements without using JSX.  
   It takes the element type, props, and children to build the virtual DOM structure.
4. **Explain how to create React nodes with JSX**  
   React nodes can be created by writing HTML-like syntax inside JavaScript using JSX.  
   These JSX elements are compiled to React.createElement() calls during build time.
5. **Define how to render JSX to DOM**  
   JSX elements are rendered to the DOM using methods like ReactDOM.render().  
   This method takes the JSX element and a DOM container as arguments.
6. **Explain how to use JavaScript expressions in JSX**  
   JavaScript expressions can be embedded in JSX by enclosing them in curly braces {}.  
   This allows dynamic data or calculations to be shown inside the UI.
7. **Explain how to use inline CSS in JSX**  
   Inline CSS in JSX is applied by passing a JavaScript object to the style attribute.  
   Property names are written in camelCase instead of the usual CSS syntax.

**// OfficeList.js**

import React from 'react';

const offices = [

{

name: 'TechHub Central',

rent: 55000,

address: '123 Innovation Street, Bangalore',

image: 'https://images.unsplash.com/photo-1562183243-f44b527227c2?auto=format&fit=crop&w=800&q=80'

},

{

name: 'Skyline Suites',

rent: 75000,

address: '456 Business Bay, Mumbai',

image: 'https://images.unsplash.com/photo-1580584126903-c17d42d4cd45?auto=format&fit=crop&w=800&q=80'

},

{

name: 'UrbanNest Workspaces',

rent: 48000,

address: '789 Startup Lane, Chennai',

image: 'https://images.unsplash.com/photo-1593642532400-2682810df593?auto=format&fit=crop&w=800&q=80'

}

];

const cardStyle = {

border: '1px solid #ddd',

borderRadius: '10px',

padding: '20px',

width: '300px',

boxShadow: '0 4px 8px rgba(0,0,0,0.1)',

backgroundColor: '#fff'

};

const imgStyle = {

width: '100%',

height: '180px',

objectFit: 'cover',

borderRadius: '8px'

};

const OfficeList = () => {

return (

<div style={{ display: 'flex', gap: '40px', flexWrap: 'wrap', justifyContent: 'center', padding: '20px' }}>

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

<div key={index} style={cardStyle}>

<img src={office.image} alt={office.name} style={imgStyle} />

<h3 style={{ margin: '10px 0 5px' }}>{office.name}</h3>

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

<p>

<strong>💰 Rent:</strong>{' '}

<span style={{ color: office.rent < 60000 ? 'red' : 'green', fontWeight: 'bold' }}>

₹{office.rent.toLocaleString()}

</span>

</p>

</div>

))}

</div>

);

};

export default OfficeList;

**//App.js**

// src/App.js

import React from 'react';

import OfficeList from './OfficeList';

function App() {

return (

<div className="App">

<h1 style={{ textAlign: 'center', margin: '30px 0', fontSize: '2rem' }}>🏙️ Office Space Rental App</h1>

<OfficeList />

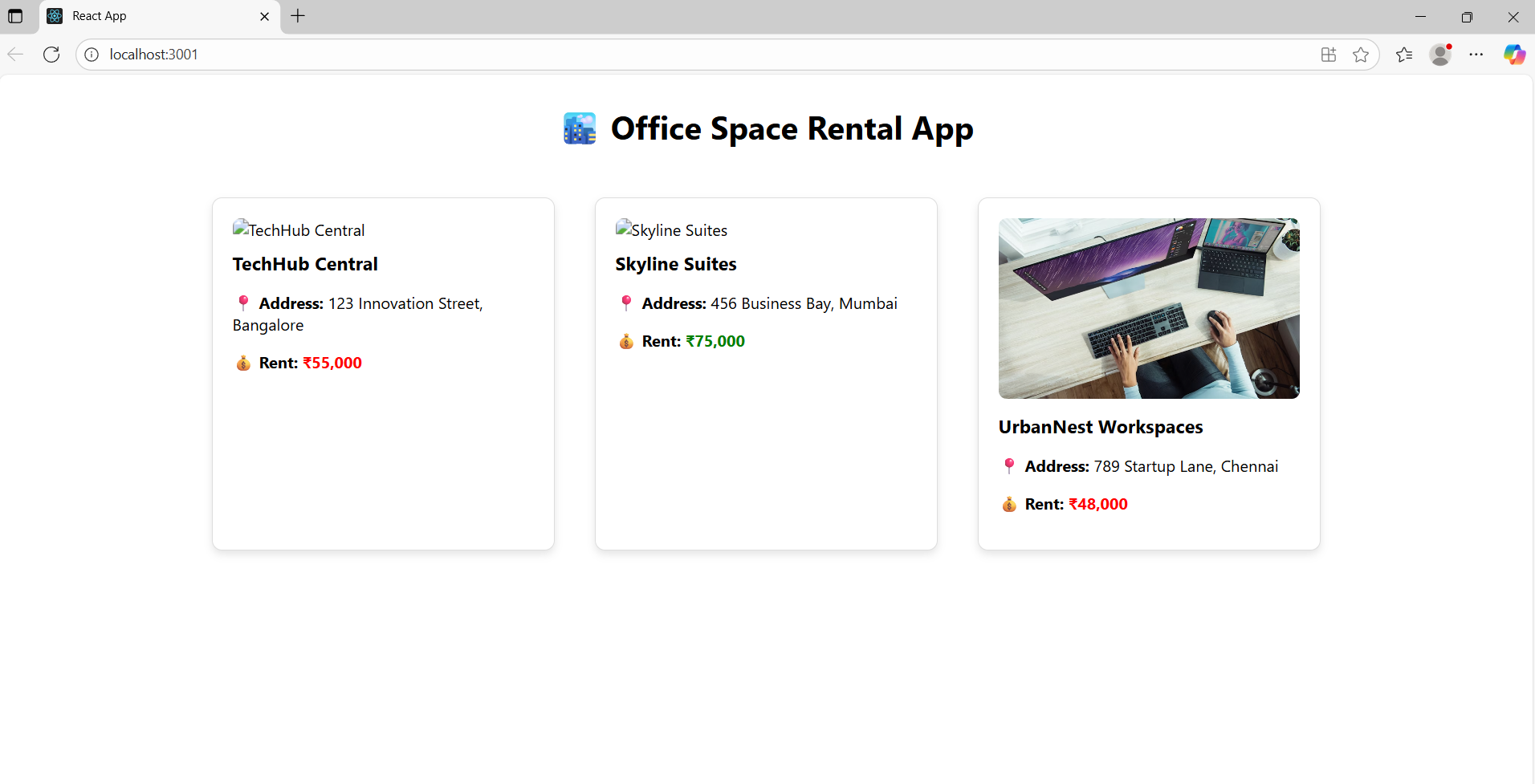
</div>

);

}

export default App;

**OUTPUT:**

****

**Exercise 3: React Events**

1. **Explain React events**  
   React events are actions triggered by user interactions like clicks, key presses, or form submissions.  
   They work similarly to DOM events but have consistent behavior across browsers.
2. **Explain about event handlers**  
   Event handlers are functions that run in response to specific events.  
   In React, these are usually passed as props like onClick or onChange.
3. **Define Synthetic event**  
   A synthetic event is a wrapper around the browser’s native event.  
   It provides a consistent interface for handling events across different browsers.
4. **Identify React event naming convention**  
   React event names use camelCase instead of lowercase.  
   For example, onclick in HTML becomes onClick in React.

**//Counter.js**

import React, { useState } from 'react';

function Counter() {

const [count, setCount] = useState(0);

const sayHello = () => {

console.log("Hello! Welcome to React events.");

};

const increment = () => {

sayHello();

setCount(prev => prev + 1);

};

const decrement = () => setCount(prev => prev - 1);

const sayWelcome = (message) => {

alert(`Welcome! ${message}`);

};

const handlePress = (event) => {

console.log("I was clicked!", event);

alert("I was clicked!");

};

return (

<div style={{ marginBottom: '40px' }}>

<h2>Counter Example</h2>

<h3>Count: {count}</h3>

<button onClick={increment}>Increment</button>{' '}

<button onClick={decrement}>Decrement</button>{' '}

<button onClick={() => sayWelcome("Glad you're here!")}>Say Welcome</button>{' '}

<button onClick={handlePress}>Click Me</button>

</div>

);

}

export default Counter;

**//CurrencyConvertor.js**

import React, { useState } from 'react';

function CurrencyConvertor() {

const [rupees, setRupees] = useState('');

const [euros, setEuros] = useState('');

const handleSubmit = () => {

const conversionRate = 0.011; // Sample conversion rate

const result = (parseFloat(rupees) \* conversionRate).toFixed(2);

setEuros(result);

};

return (

<div>

<h2>Currency Convertor</h2>

<input

type="number"

value={rupees}

onChange={(e) => setRupees(e.target.value)}

placeholder="Enter INR"

/>

<button onClick={handleSubmit}>Convert</button>

{euros && <p>💶 EUR: €{euros}</p>}

</div>

);

}

export default CurrencyConvertor;

**//App.js**

import React from 'react';

import Counter from './components/Counter';

import CurrencyConvertor from './components/CurrencyConvertor';

function App() {

return (

<div style={{ padding: '30px', fontFamily: 'Arial, sans-serif' }}>

<h1>🎯 React Event Handling Examples</h1>

<Counter />

<CurrencyConvertor />

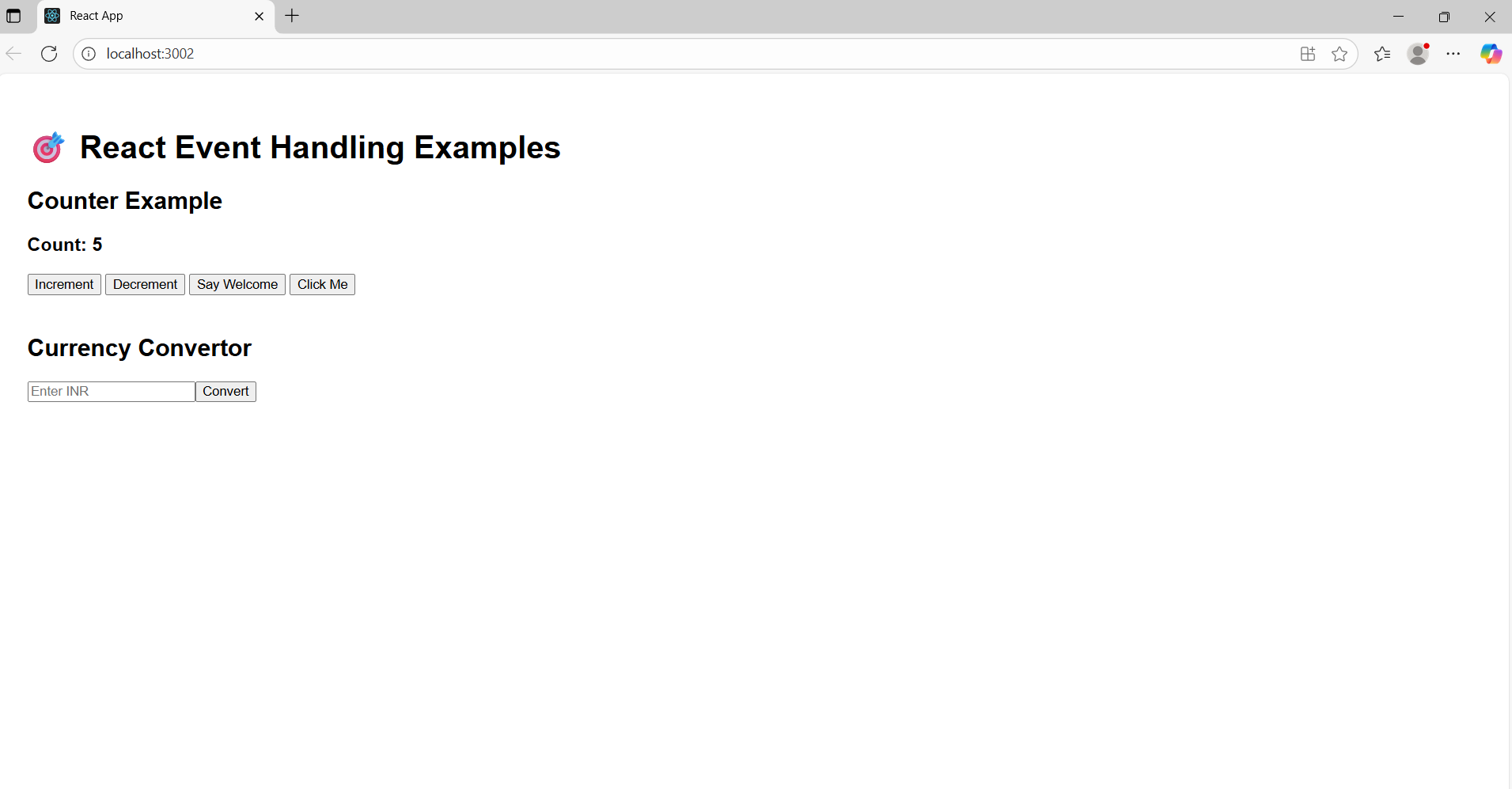
</div>

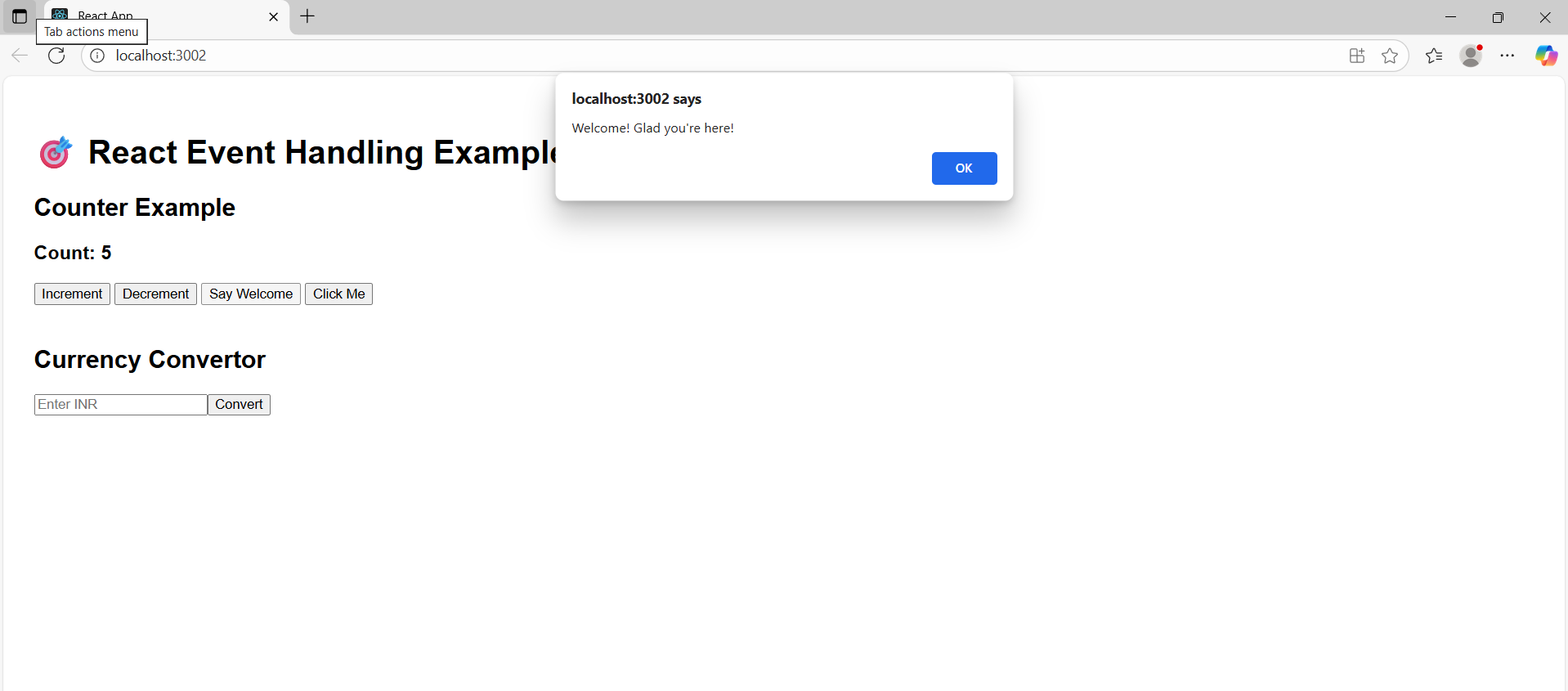
);

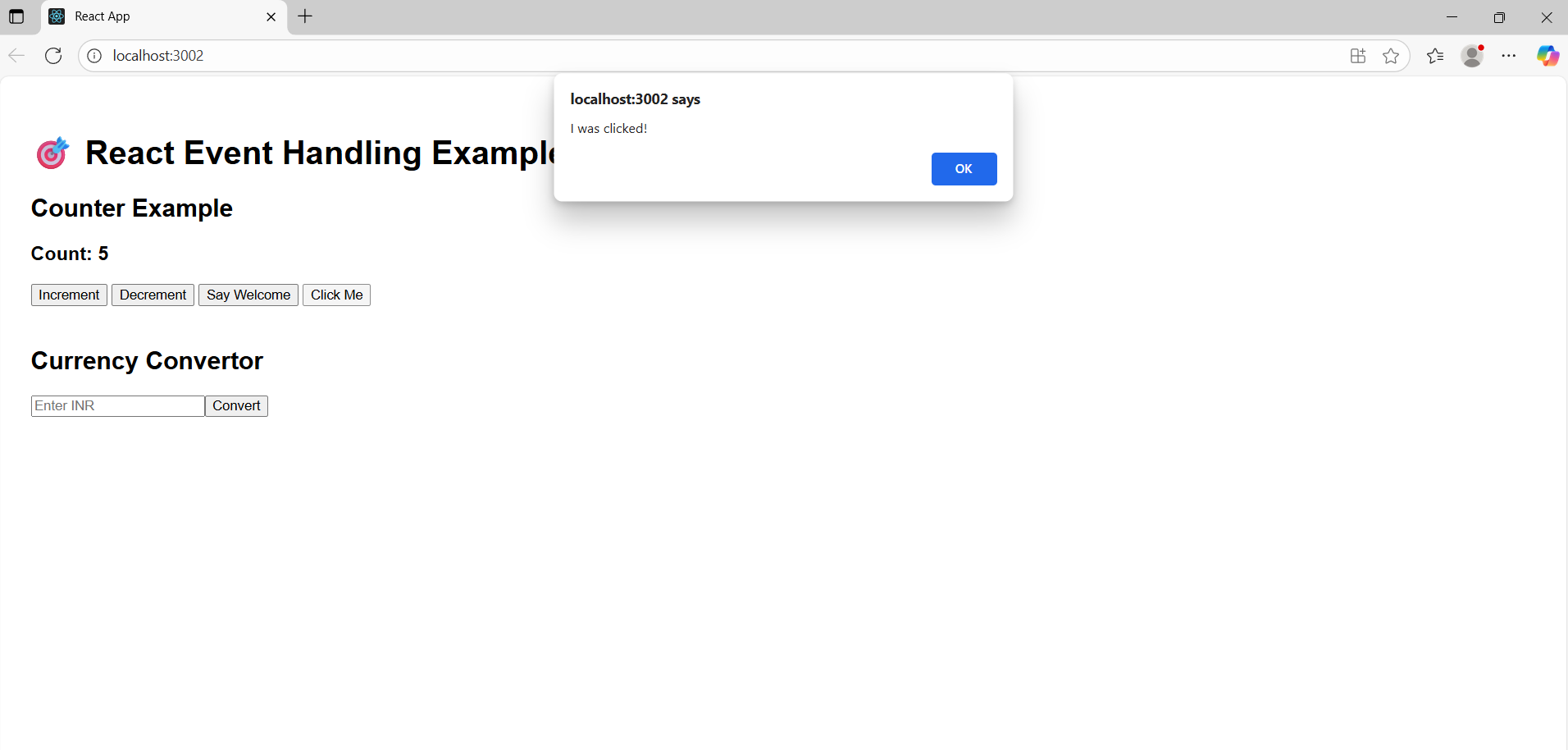
}

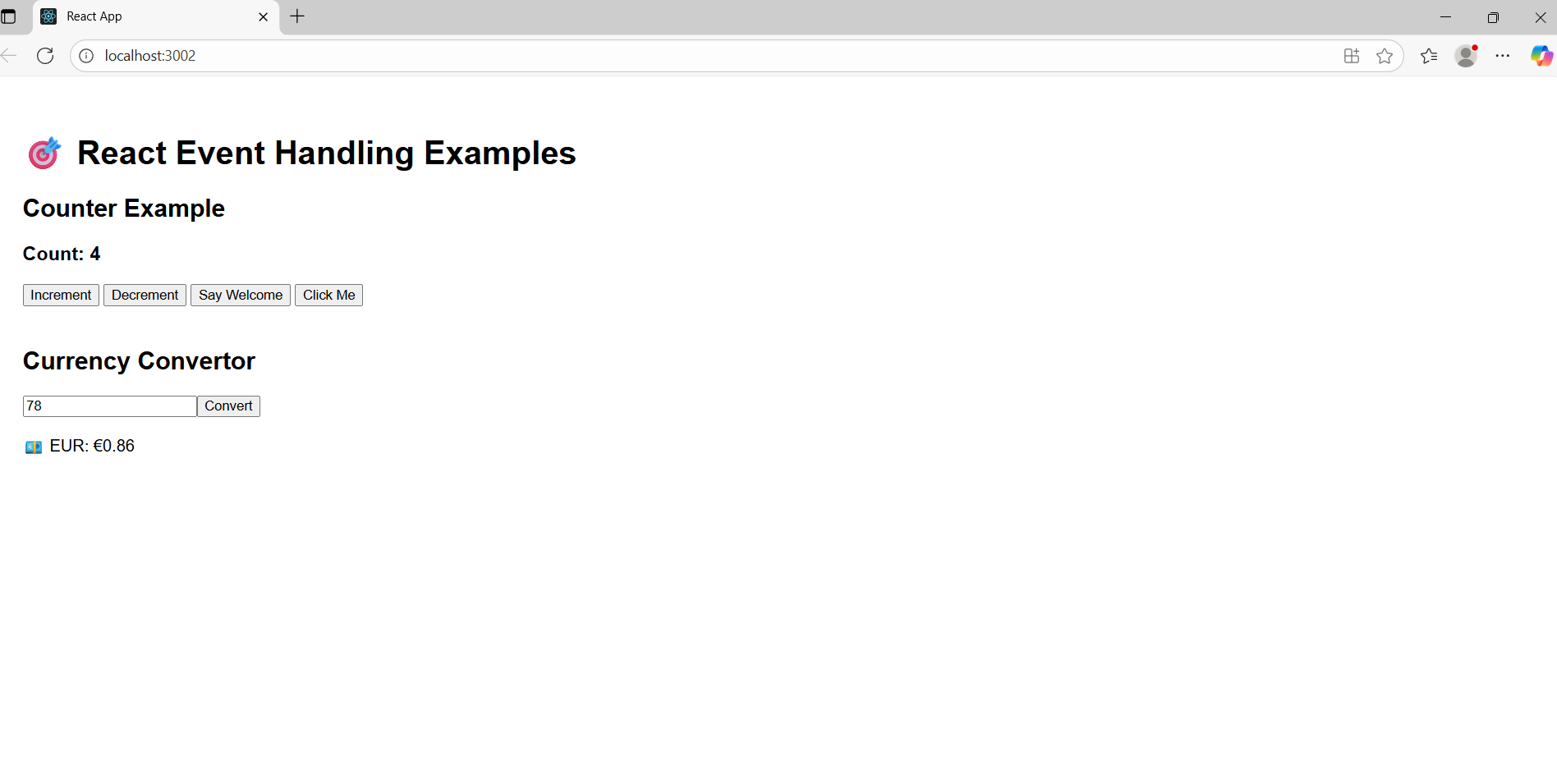
export default App;

**OUTPUT:**



****

****

****

**Exercise 4: Rendering**

1. **Explain about conditional rendering in React**Conditional rendering is used to display different UI elements based on certain conditions.  
   It works by using JavaScript expressions like if, ternary operators, or logical && inside JSX.
2. **Define element variables**Element variables store JSX elements in variables, making it easier to decide what to render.  
   They can be assigned different JSX based on conditions before being returned in render().
3. **Explain how to prevent components from rendering**Components can be prevented from rendering by returning null instead of JSX.  
   This effectively hides the component without affecting the rest of the UI.

**//GuestPage.js**

import React from 'react';

import FlightDetails from './FlightDetails';

const GuestPage = ({ onLogin }) => {

return (

<div>

<h2>Welcome, Guest! 👋</h2>

<FlightDetails />

<button onClick={onLogin} style={{ marginTop: '20px' }}>Login to Book Tickets</button>

</div>

);

};

export default GuestPage;

**// UserPage.js**

import React from 'react';

import FlightDetails from './FlightDetails';

const UserPage = ({ onLogout }) => {

return (

<div>

<h2>Welcome, Harini! You’re logged in ✅</h2>

<FlightDetails showBooking={true} />

<button onClick={onLogout} style={{ marginTop: '20px' }}>Logout</button>

</div>

);

};

export default UserPage;

**// FlightDetails.js**

import React from 'react';

const flights = [

{ id: 'AI101', destination: 'Delhi', time: '08:00 AM' },

{ id: '6E222', destination: 'Mumbai', time: '11:30 AM' },

{ id: 'SG303', destination: 'Chennai', time: '03:45 PM' }

];

const FlightDetails = ({ showBooking }) => {

return (

<div>

<h3>Flight Details:</h3>

<ul>

{flights.map(flight => (

<li key={flight.id}>

✈️ {flight.id} to {flight.destination} at {flight.time}

{showBooking && (

<button style={{ marginLeft: '10px' }}>Book Now</button>

)}

</li>

))}

</ul>

</div>

);

};

export default FlightDetails;

**//App.js**

import React, { useState } from 'react';

import GuestPage from './components/GuestPage';

import UserPage from './components/UserPage';

function App() {

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

const handleLogin = () => setIsLoggedIn(true);

const handleLogout = () => setIsLoggedIn(false);

let pageContent;

if (isLoggedIn) {

pageContent = <UserPage onLogout={handleLogout} />;

} else {

pageContent = <GuestPage onLogin={handleLogin} />;

}

return (

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

<h1 style={{ textAlign: 'center' }}>✈️ Ticket Booking App</h1>

{pageContent}

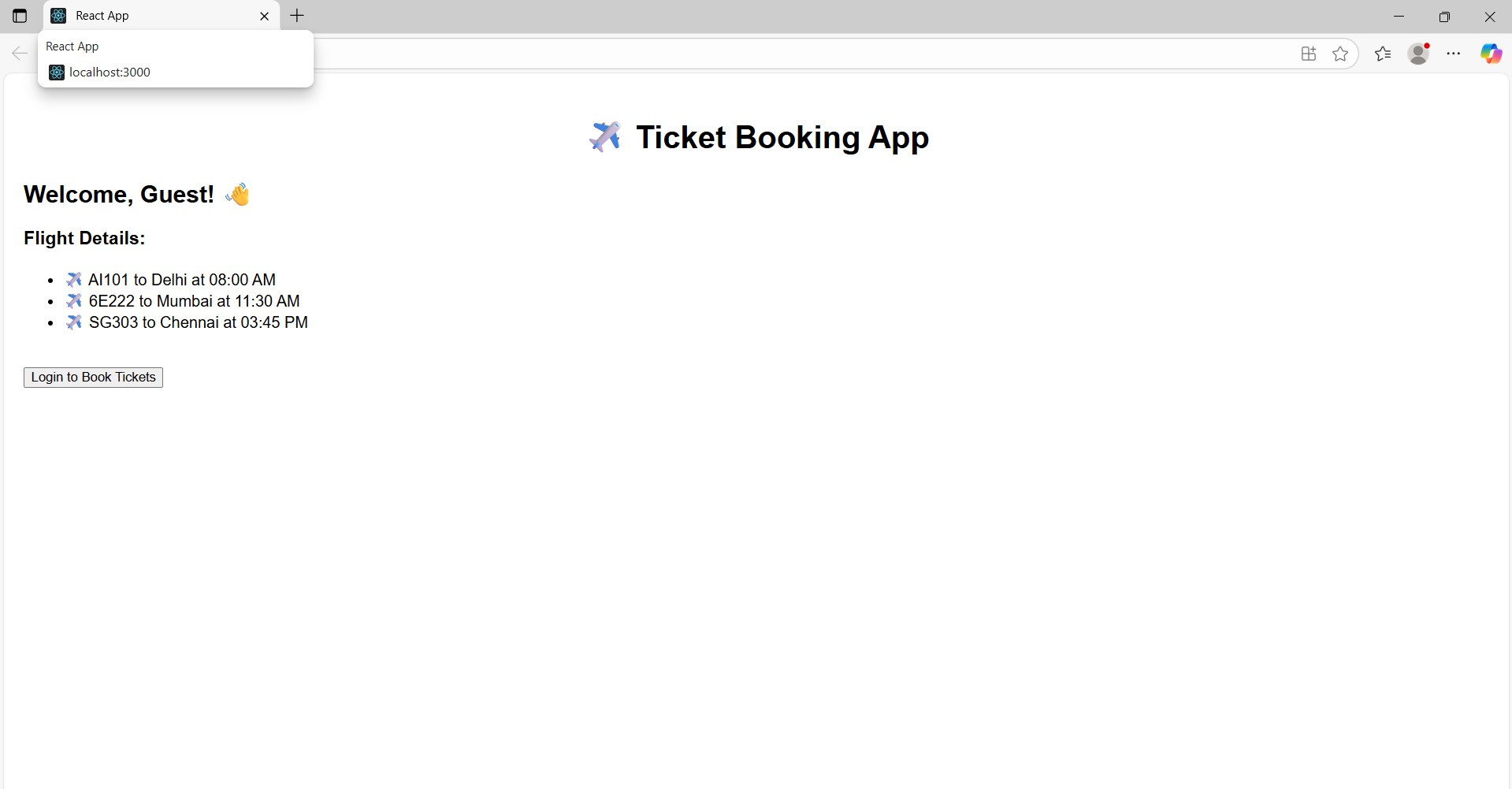
</div>

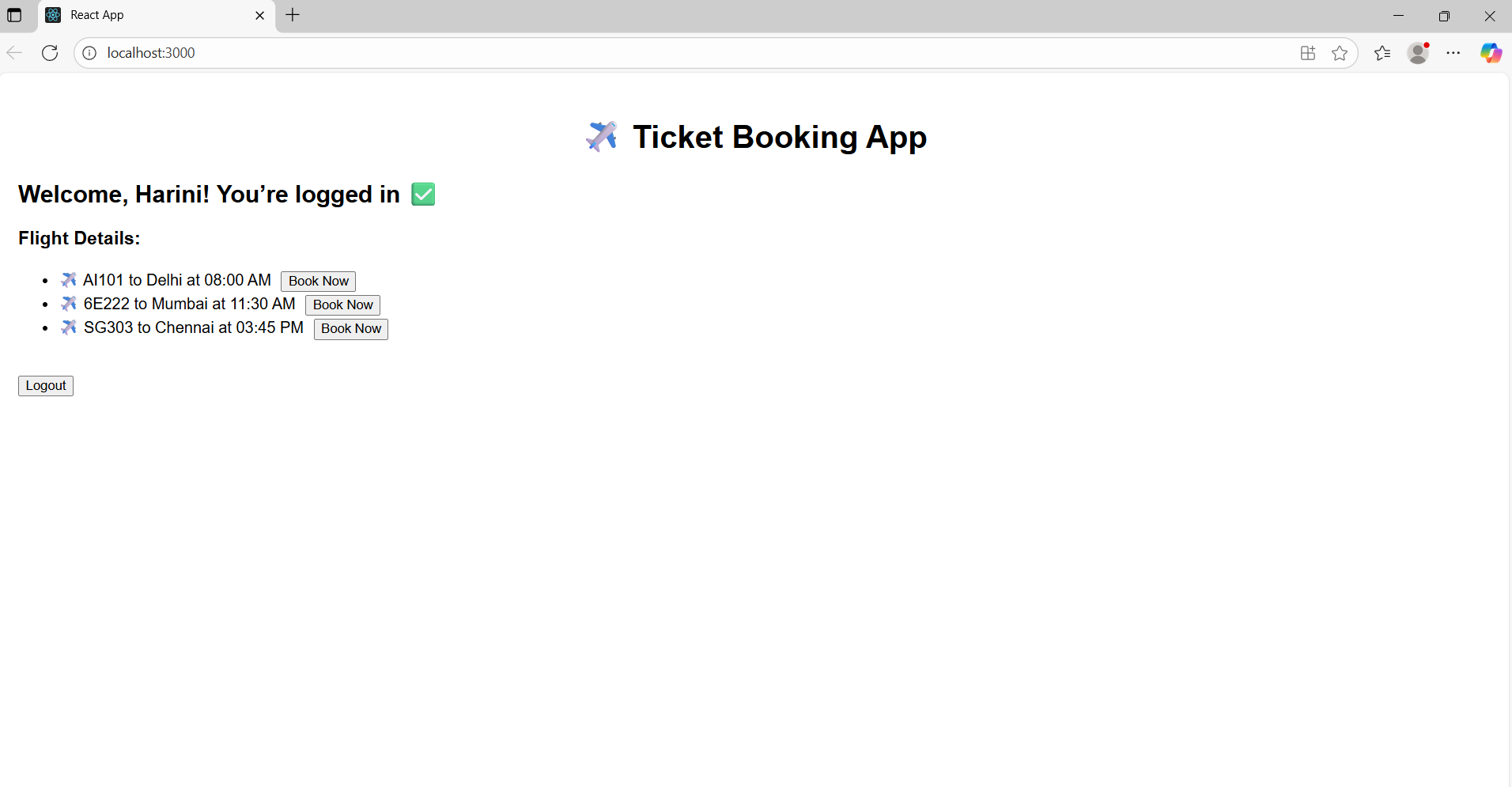
);

}

export default App;

**OUTPUT:**

****

****

**Exercise 5: Conditional Rendering**

1. **Explain various ways of conditional rendering**  
   Conditional rendering can be done using if statements, ternary operators, logical && operators, and element variables.  
   These methods help show or hide components based on certain conditions.
2. **Explain how to render multiple components**  
   Multiple components can be rendered together by returning them inside a single parent element like a <div> or using fragments <> </>.  
   This helps build complex UIs by combining smaller reusable components.
3. **Define list component**  
   A list component is used to display a collection of similar items by looping through data.  
   It typically uses the map() function to create a list of JSX elements.
4. **Explain about keys in React applications**  
   Keys are special string attributes used to identify which items in a list have changed, been added, or removed.  
   They help React optimize rendering and keep components in sync with data.
5. **Explain how to extract components with keys**  
   You can create a new child component to render each list item and pass a unique key as a prop.  
   This makes the code cleaner and ensures each item is properly tracked by React.
6. **Explain React Map, map() function**  
   The map() function is used to iterate over arrays and transform each item into a JSX element.  
   In React, it’s commonly used to dynamically render lists of components.

**//BlogDetails.js**

import React from 'react';

const BlogDetails = () => {

const showDetails = true;

return (

<>

<h2>📰 Blog Posts</h2>

{showDetails && (

<ul>

<li>React vs Vue: A Developer’s Take</li>

<li>Best Coding Practices in 2025</li>

<li>State Management with Redux Toolkit</li>

</ul>

)}

</>

);

};

export default BlogDetails;

**//BookDetails.js**

import React from 'react';

const books = [

{ id: 1, title: 'Eloquent JavaScript', show: true },

{ id: 2, title: 'You Don’t Know JS', show: true },

{ id: 3, title: 'Clean Code', show: false }

];

const BookDetails = () => {

return (

<div>

<h2>📚 Book Recommendations</h2>

<ul>

{books.map(book => book.show && <li key={book.id}>{book.title}</li>)}

</ul>

</div>

);

};

export default BookDetails;

**//CourseDetails.js**

import React from 'react';

const CourseDetails = () => {

const isEnrolled = false;

let message;

if (isEnrolled) {

message = <p>✅ You’re enrolled in React Advanced!</p>;

} else {

message = <p>⏳ Enroll now to unlock all React lessons!</p>;

}

return (

<div>

<h2>🎓 Course Details</h2>

{message}

</div>

);

};

export default CourseDetails;

**//App.js**

import React, { useState } from 'react';

import BlogDetails from './components/BlogDetails';

import BookDetails from './components/BookDetails';

import CourseDetails from './components/CourseDetails';

function App() {

const [view, setView] = useState('blog'); // 'book' | 'course'

return (

<div style={{ padding: '20px', fontFamily: 'sans-serif' }}>

<h1>📝 Blogger App</h1>

<div style={{ marginBottom: '20px' }}>

<button onClick={() => setView('blog')}>Show Blogs</button>

<button onClick={() => setView('book')}>Show Books</button>

<button onClick={() => setView('course')}>Show Courses</button>

</div>

{/\* 1️⃣ Using If-Else \*/}

{view === 'blog' ? (

<BlogDetails />

) : view === 'book' ? (

<BookDetails />

) : (

<CourseDetails />

)}

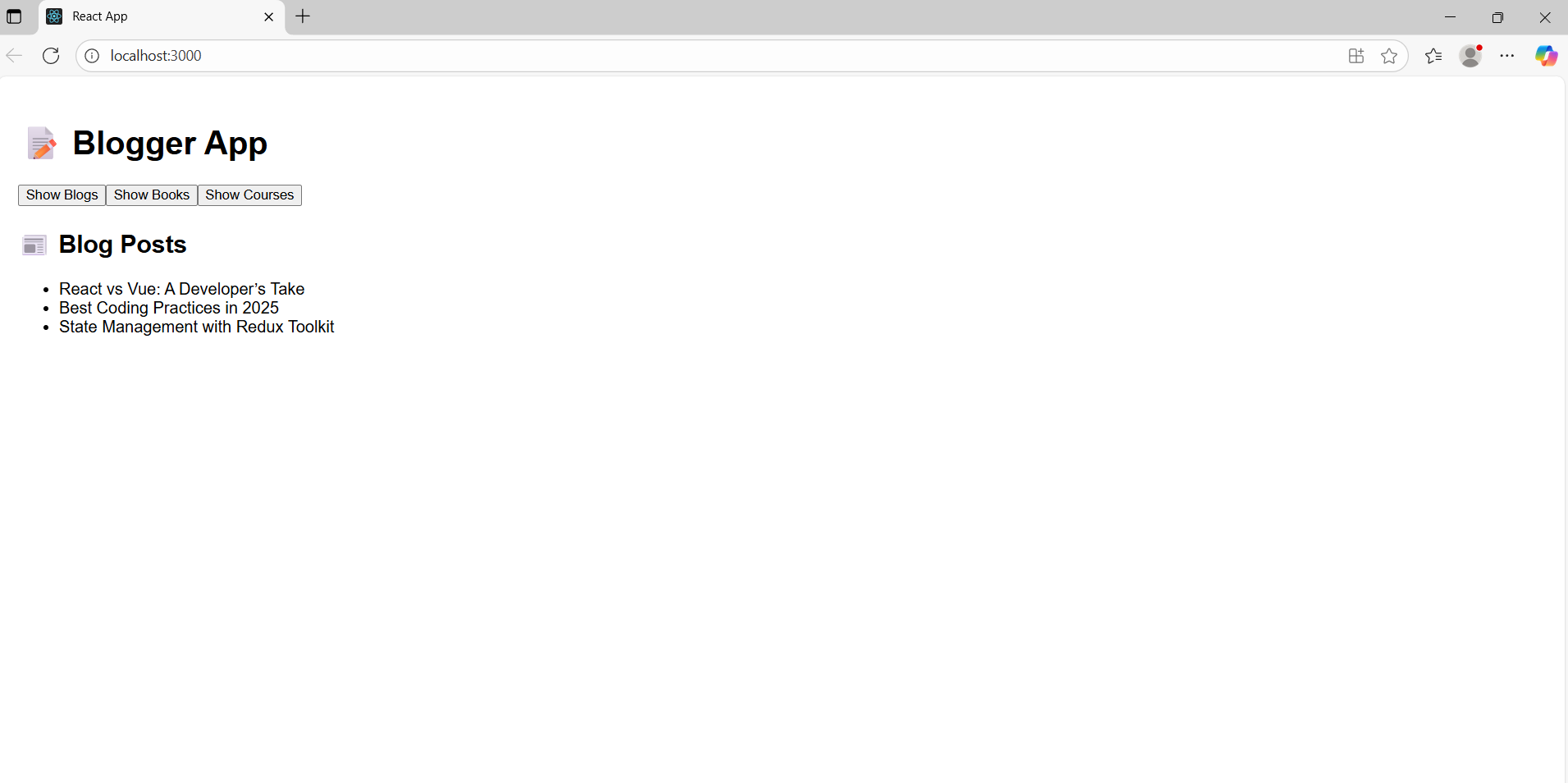
</div>

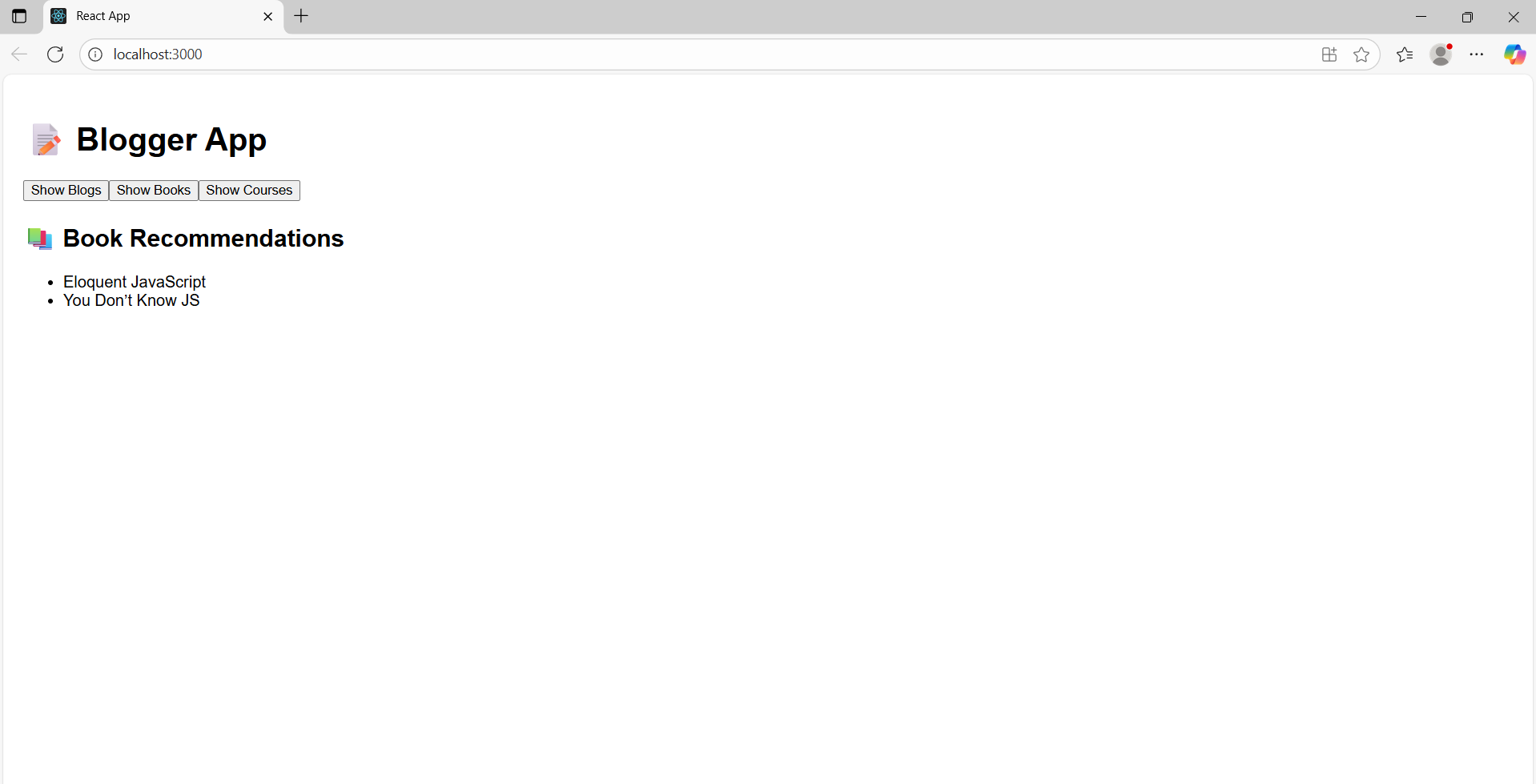
);

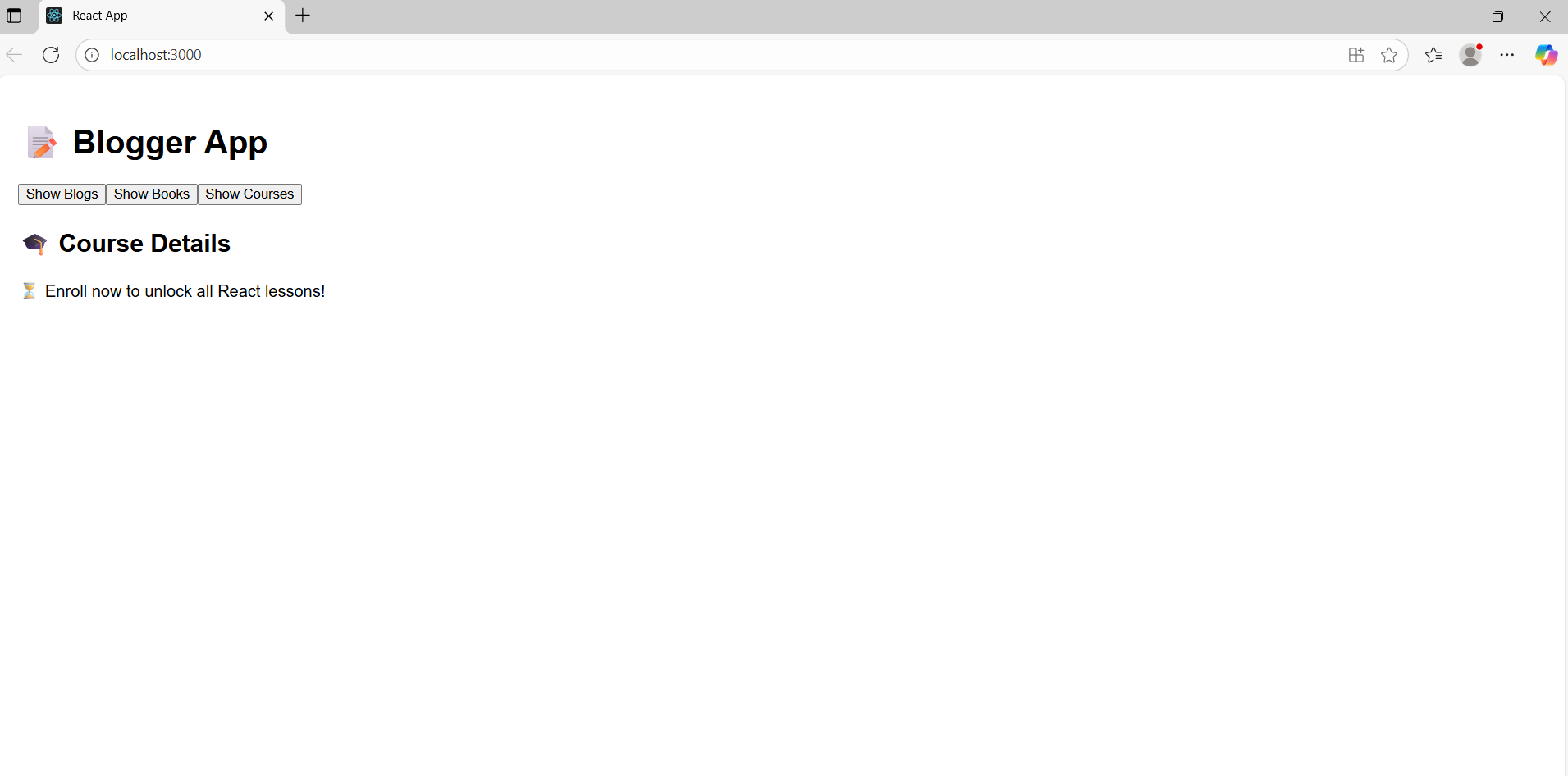
}

export default App;

**OUTPUT:**

****

****

****

**NAME: HARINI G  
SUPERSET ID: 6385037**