**Mandatory Question for React Week 7**

**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:**



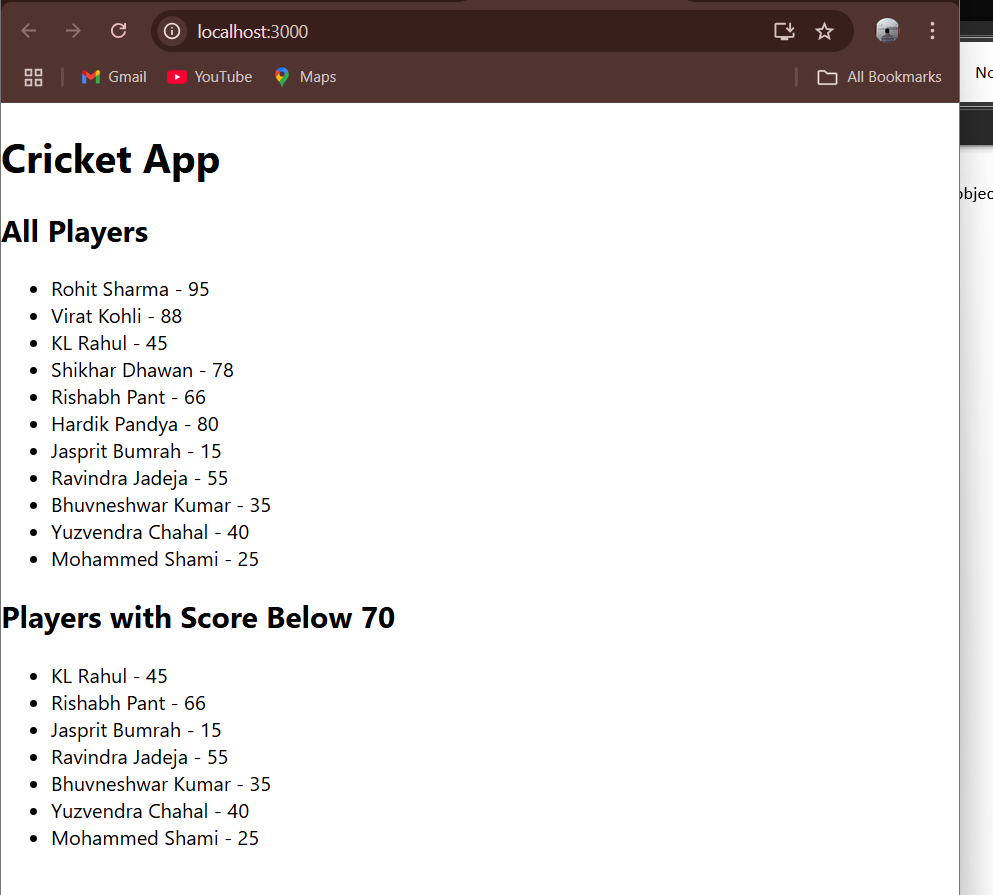
**. ES6 Features Overview**

ECMAScript 6 (ES6) introduced major improvements to JavaScript that make code cleaner, more powerful, and easier to read.

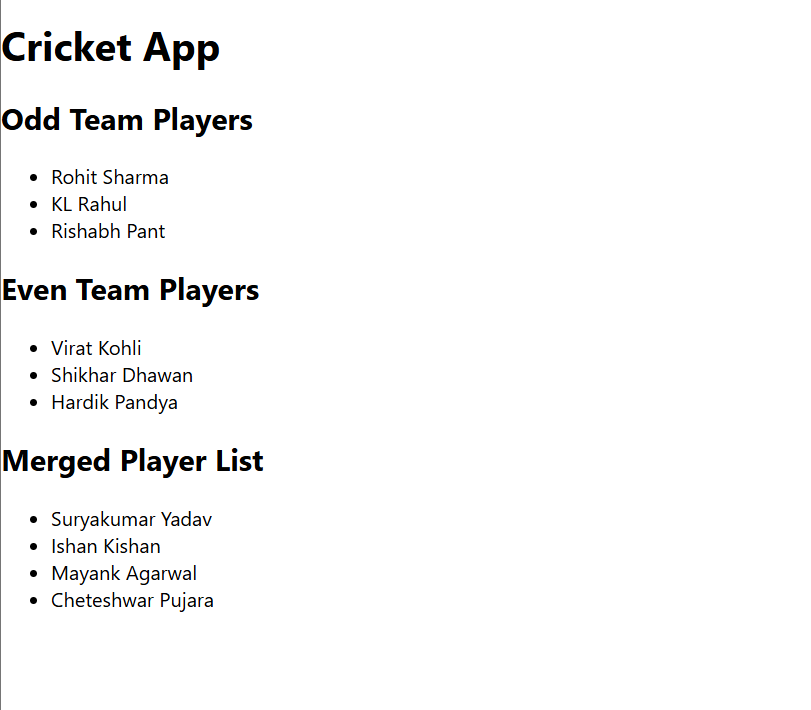
**Key ES6 Features in This Lab**

| **Feature** | **Description** | **Example** |
| --- | --- | --- |
| **let** | Declares block-scoped variables (cannot be redeclared in the same scope) | let score = 50; |
| **const** | Declares variables whose value cannot be reassigned | const pi = 3.14; |
| **Arrow Functions** | Shorter syntax for functions, this binding is lexical | const add = (a,b) => a+b; |
| **map()** | Creates a new array by applying a function to each element | arr.map(x => x\*2) |
| **filter()** | Returns array elements that pass a condition | arr.filter(x => x>5) |
| **Destructuring** | Extracts values from arrays/objects into variables | const [a,b] = [1,2] |
| **Spread Operator (...)** | Expands elements from arrays/objects | const arr3 = [...arr1, ...arr2] |
| **Set** | Stores unique values | new Set([1,1,2]) → {1,2} |
| **Map** | Stores key-value pairs | let m = new Map([['a',1]]) |

Output:-  
  
When Flag=True;



When Flag=False;



**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:**





-  
**Theory**

**1. JSX**

JSX (**JavaScript XML**) is a syntax extension for JavaScript used in React to describe the UI structure. It looks like HTML but works inside JavaScript files.

Example:

const element = <h1>Hello World</h1>;

**2. ECMA Script**

ECMA Script is the standardized specification for JavaScript. ES6 introduced features like let, const, arrow functions, classes, destructuring, map, and more.

**3. React.createElement()**

Before JSX, UI elements in React were created using:

React.createElement('h1', null, 'Hello World');

JSX is syntactic sugar for React.createElement().

**4. Creating React Nodes with JSX**

JSX allows creating elements with HTML-like syntax directly in JavaScript:

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

**5. Rendering JSX to DOM**

React DOM renders JSX to the browser:

ReactDOM.render(heading, document.getElementById('root'));

**6. Using JavaScript Expressions in JSX**

JavaScript expressions can be embedded in JSX using {}:

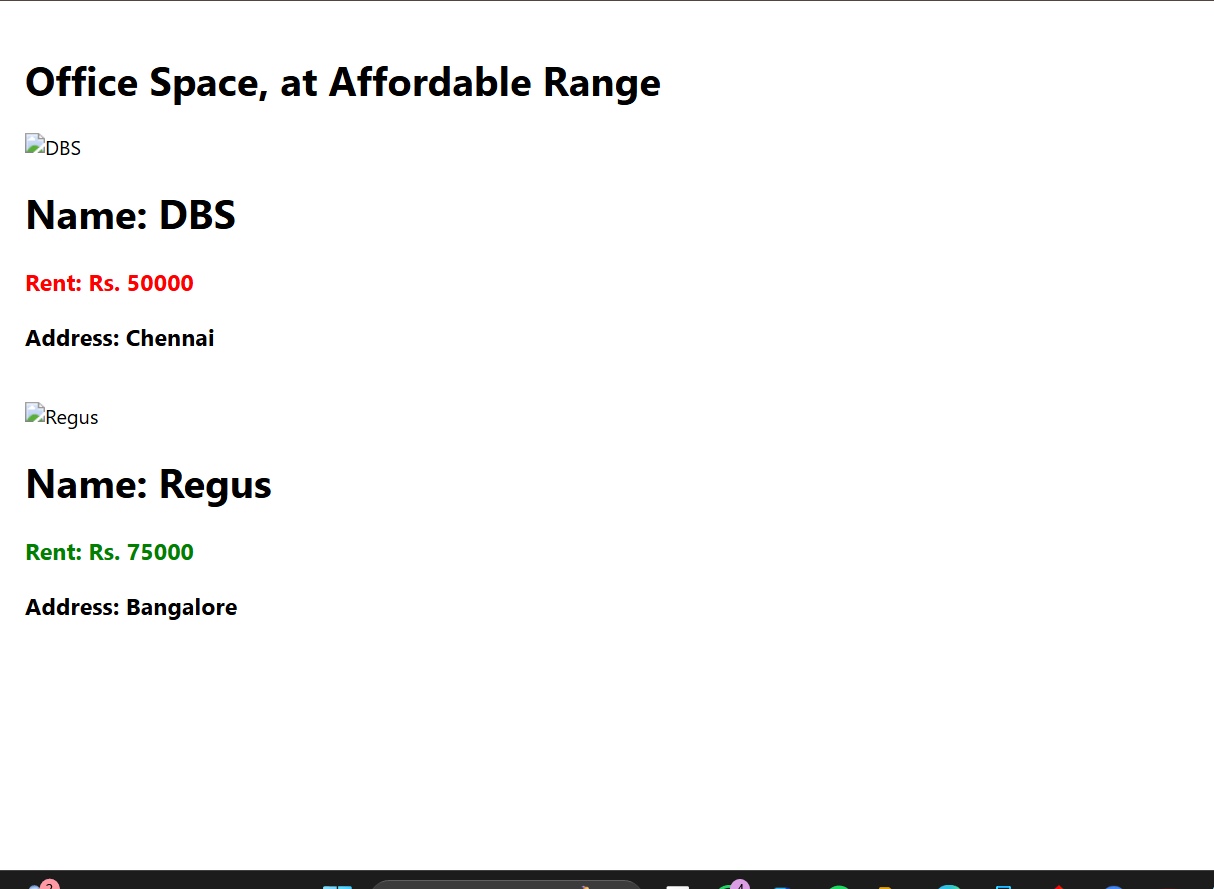
<h1>{office.Name}</h1>

**7. Using Inline CSS in JSX**

CSS can be applied inline using style={{}} or class names:

<h3 style={{ color: 'red' }}>Rent: Rs. 50000</h3>

Output:-



**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 concept in React applications
* Use this keyword
* Use synthetic event

**Prerequisites**

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

* Node.js
* NPM
* Visual Studio Code

**Notes**

Estimated time to complete this lab: **90 minutes.**

Create a React Application “eventexamplesapp” to handle various events of the form elements in HTML.

1. Create “Increment” button to increase the value of the counter and “Decrement” button to decrease the value of the counter. The “Increase” button should invoke multiple methods.
   1. To increment the value
   2. Say Hello followed by a static message.



1. Create a button “Say Welcome” which invokes the function which takes “welcome” as an argument.



1. Create a button which invokes synthetic event “OnPress” which display “I was clicked”



Create a “CurrencyConvertor” component which will convert the Indian Rupees to Euro when the Convert button is clicked.

Handle the Click event of the button to invoke the handleSubmit event and handle the conversion of the euro to rupees.



**React Events**

React events are interactions or actions triggered by the user or the browser, such as clicks, form submissions, mouse movements, or keyboard presses. React uses its own event system that is consistent across all browsers, making event handling predictable and reliable.

**Event Handlers**

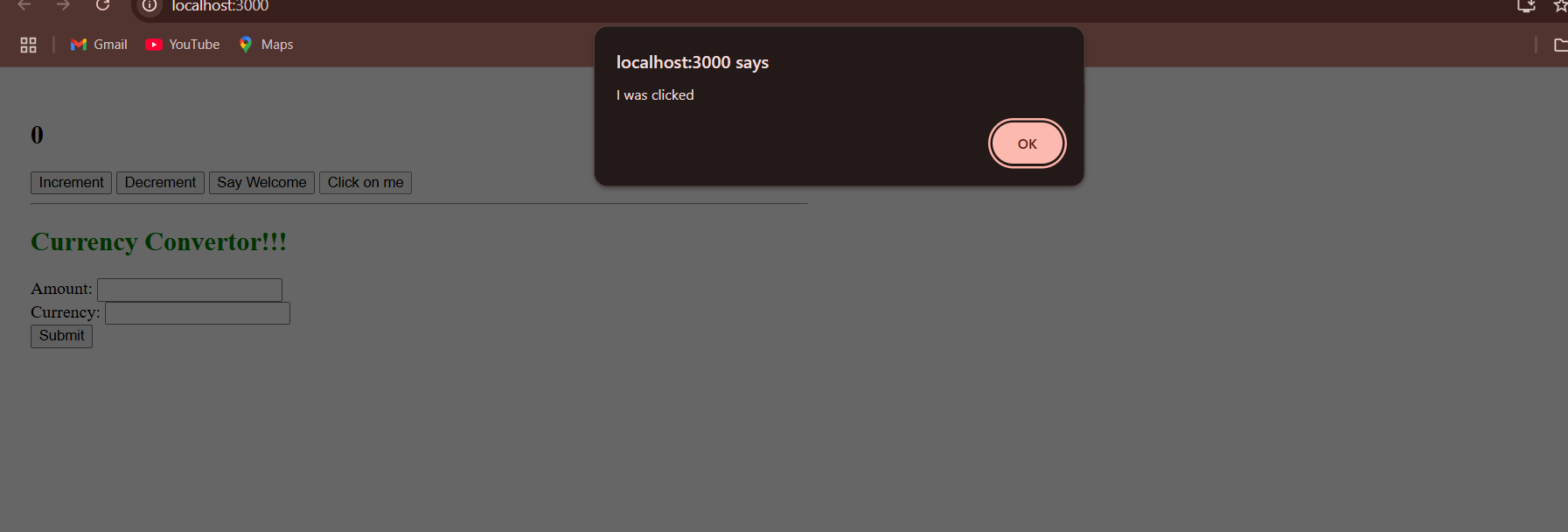
Event handlers are functions that are executed when a specific event occurs. In React, they are used to respond to user interactions and update the component’s state or perform other operations. They are passed as properties to elements and are triggered automatically when the corresponding event takes place.

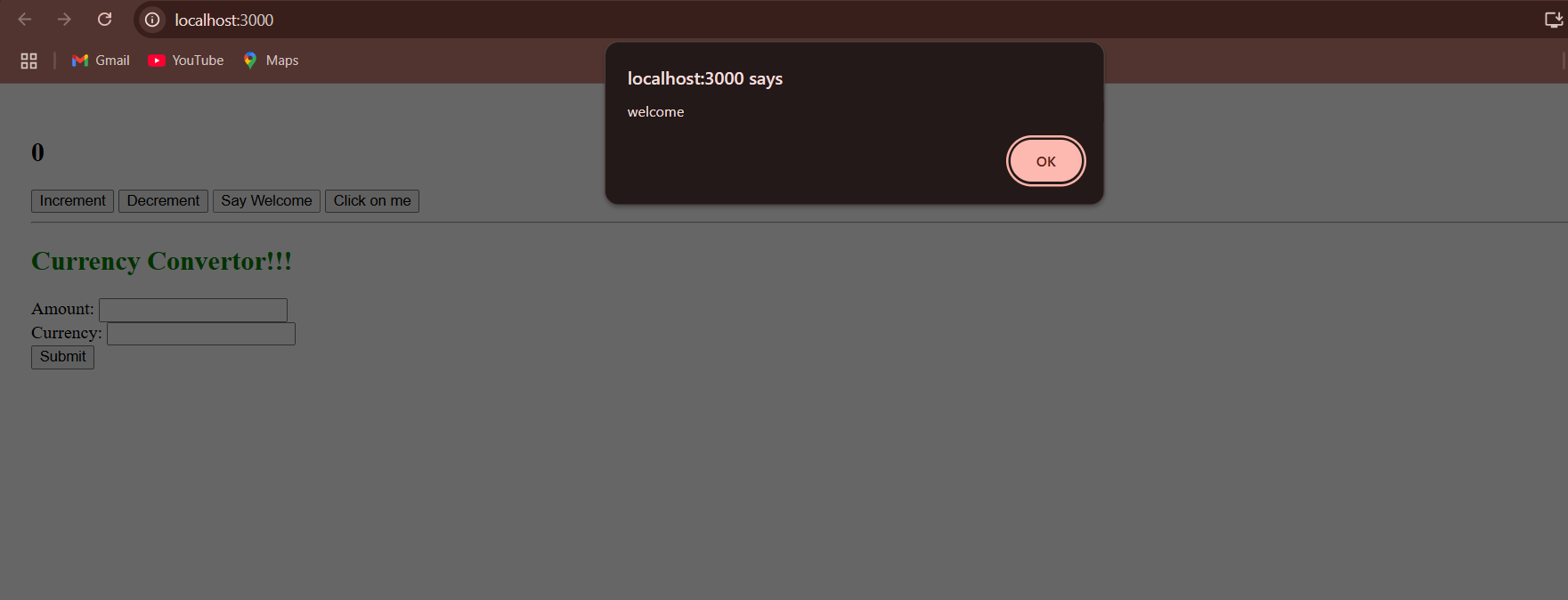
**Synthetic Event**

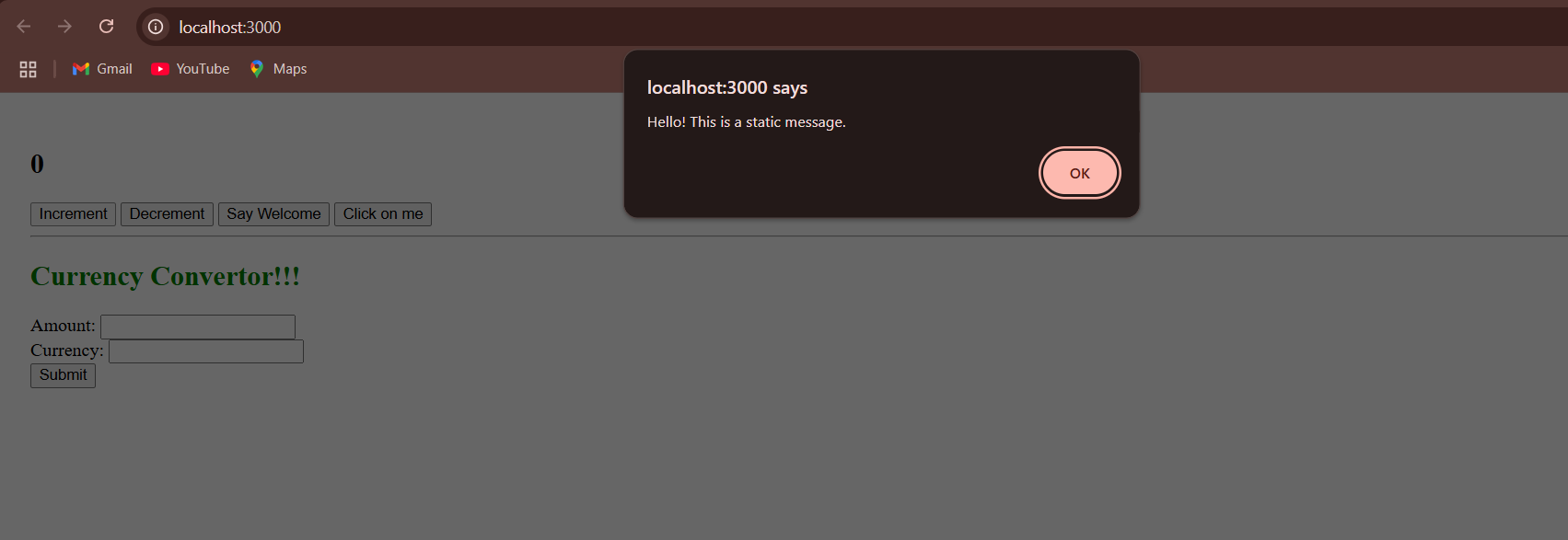
A Synthetic Event is React’s cross-browser wrapper around the native browser event. It normalizes the event object, ensuring that the event has the same properties and methods in all browsers. Synthetic events are lightweight and are part of React’s optimization for event handling.

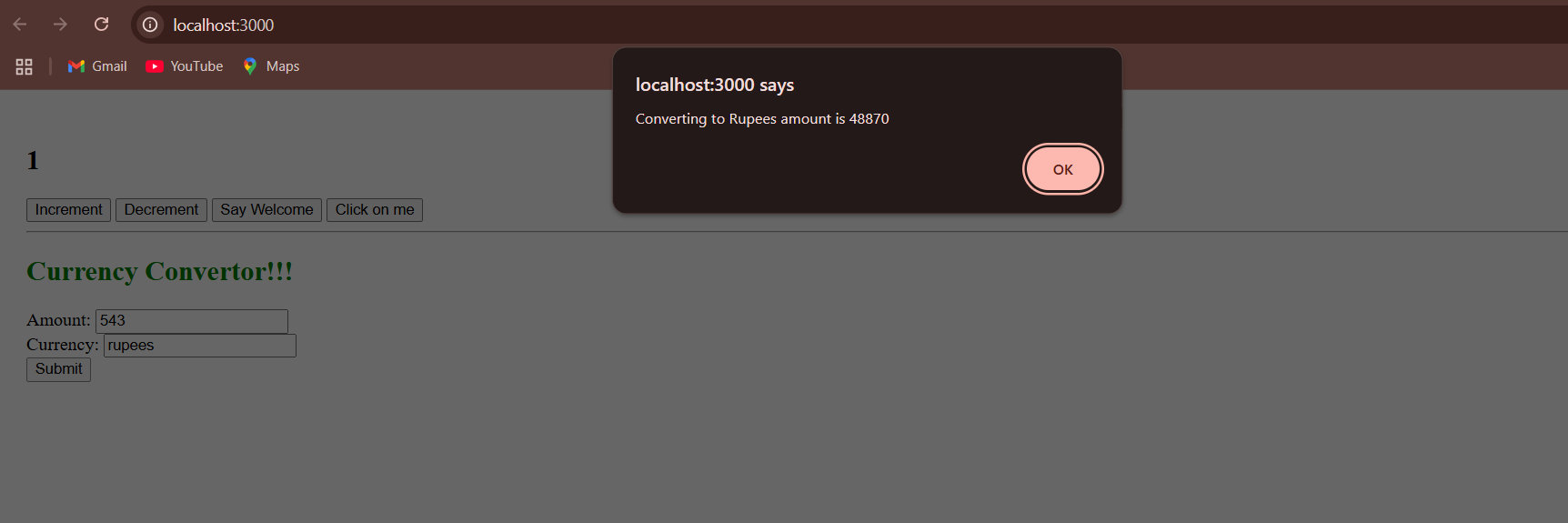
**React Event Naming Convention**

In React, event names follow a camelCase naming convention rather than lowercase. For example, onClick is used instead of onclick. This helps distinguish between native HTML attributes and React’s synthetic event system.  
  
  
Output:









**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**

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 “ticketbookingapp” where the guest user can browse the page where the flight details are displayed whereas the logged in user only can book tickets.

The Login and Logout buttons should accordingly display different pages. Once the user is logged in the User page should be displayed. When the user clicks on Logout, the Guest page should be displayed.





**Hint:**







**Conditional Rendering in React**

Conditional rendering means displaying different UI based on certain conditions, just like using if statements in JavaScript. In React, it allows components to adapt what is shown to the user depending on state or props.

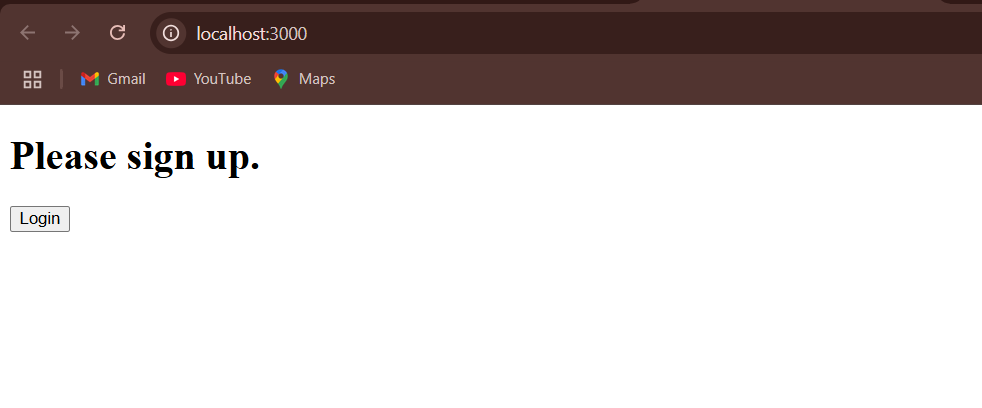
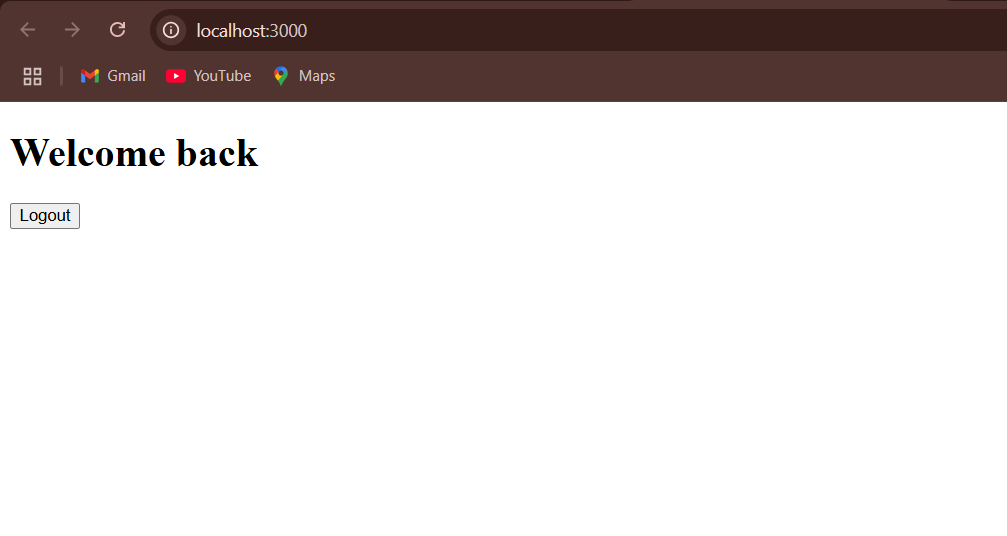
**Element Variables**

React lets you store elements in variables, then decide later whether to render them. This can make conditional rendering cleaner and avoid repeating JSX.

**Preventing Components from Rendering**

You can stop a component from rendering by:

* Returning null in its render method (or function body for functional components)
* Not including it in JSX when conditions are false

Output:-  
   
  
  
  


**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**

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 App named “bloggerapp” in with 3 components.

1. Book Details
2. Blog Details
3. Course Details

Implement this with as many ways possible of Conditional Rendering.



**Hint:**







**Theory**

**1. Various Ways of Conditional Rendering in React**

* **Using if/else statements** in the component before returning JSX.
* **Using logical && operator** to render only if a condition is true.
* **Using ternary operator** condition ? <A/> : <B/>.
* **Using variable assignment** to decide what to render, then returning that variable.

**2. Rendering Multiple Components**

You can render multiple components by including them in JSX like:

<>

<Component1 />

<Component2 />

</>

or by placing them side by side in a container <div> or <section>.

**3. List Component**

A list component in React displays data by iterating over an array (usually with .map()), generating an element for each item.

**4. Keys in React**

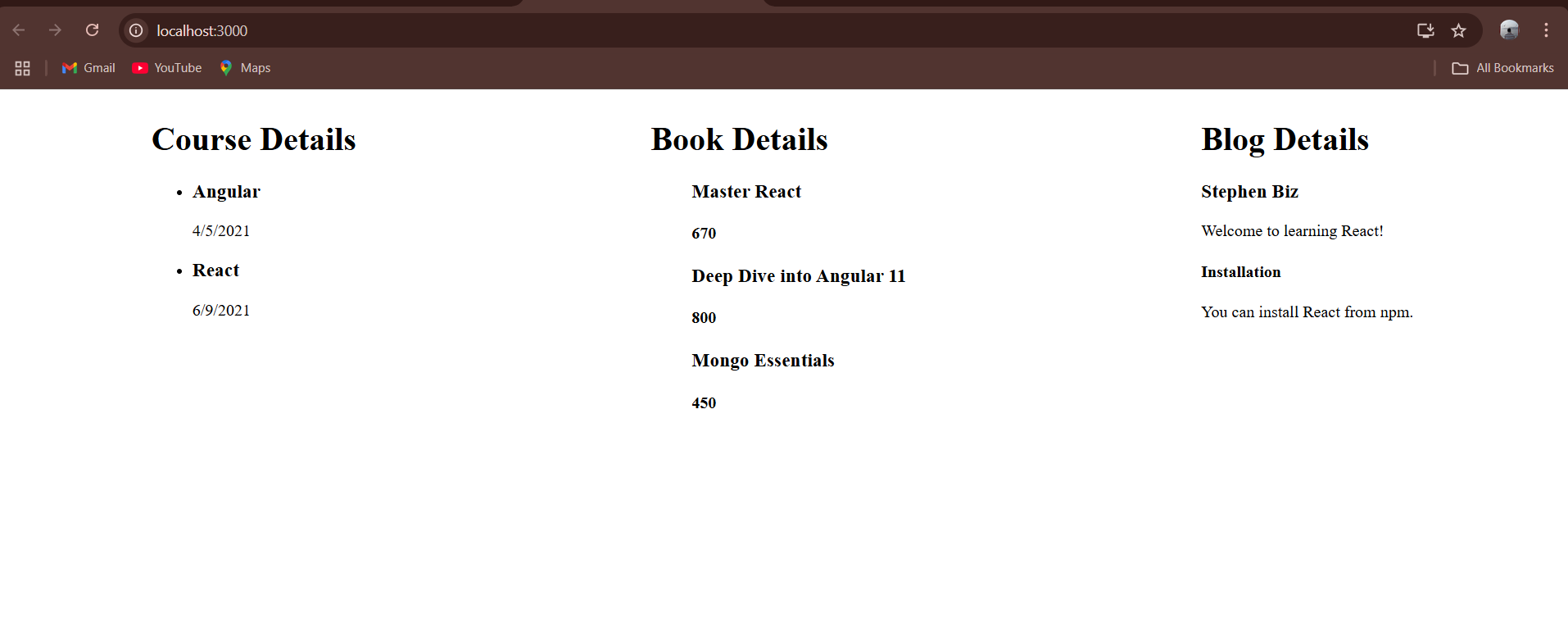
Keys are unique identifiers for list items.  
They help React efficiently re-render lists by tracking changes, additions, and deletions.

**5. Extracting Components with Keys**

When you break a list into separate components, pass the key to the top-level element inside that component so React can track it.

**6. React map() Function**

The .map() function in JavaScript creates a new array by transforming each element in the source array.  
In React, it’s often used to generate a list of JSX elements.

Output:  
  


*Thank You<>*