**React 11**

### ****1. Explain React Events****

In React, **events** are handled similarly to native HTML DOM events, but with some key differences:

React wraps native events in a **SyntheticEvent** system for **cross-browser compatibility**.

Events are attached to elements using **JSX attributes**, like onClick, onChange, etc.

#### Example:

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

### ****2. Explain Event Handlers****

**Event handlers** are functions that are triggered when a specific event occurs, such as a click, keypress, or form submission.

#### Example:

function handleClick() {

alert('Button clicked!');

}

return <button onClick={handleClick}>Click</button>;

We can also define handlers **inline**:

<button onClick={() => alert('Clicked')}>Click</button>

### ****3. Define Synthetic Event****

A **SyntheticEvent** in React is a wrapper around the browser’s native event system. It normalizes events so that they behave consistently across all browsers.

Works identically across different browsers.

React reuses SyntheticEvent objects through pooling for performance optimization (in older React versions).

#### Example:

function handleInput(event) {

console.log(event); // SyntheticEvent

console.log(event.target.value); // Actual input value

}

SyntheticEvent wraps native events like MouseEvent, KeyboardEvent, InputEvent, etc.

### ****4. React Event Naming Convention****

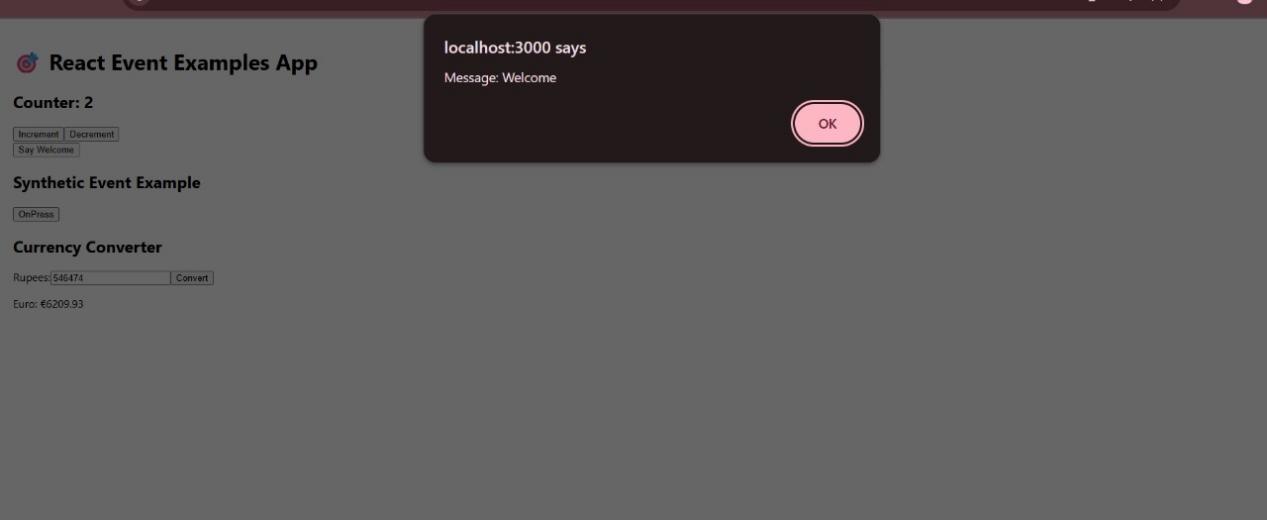
React uses **camelCase** naming for event handlers instead of lowercase HTML style.

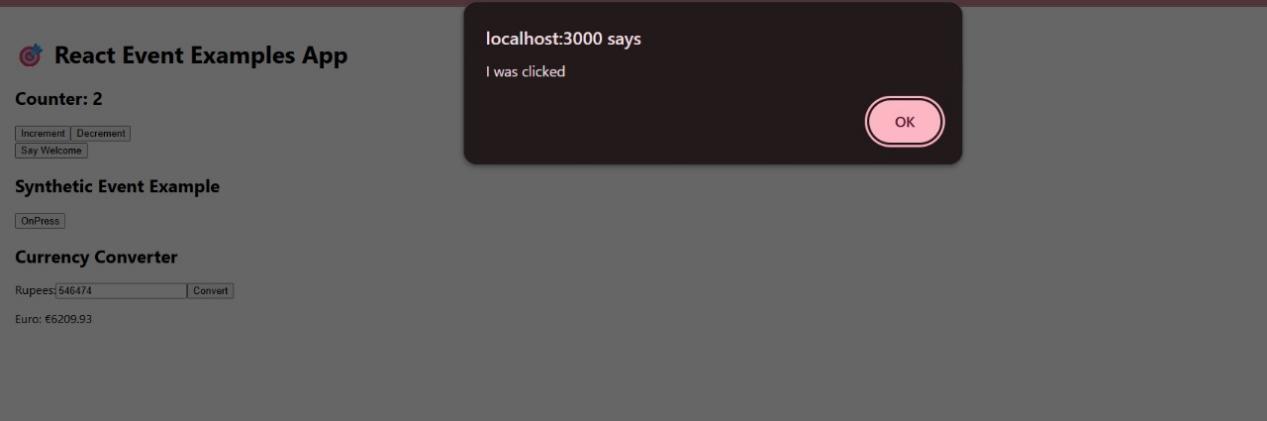
| **HTML** | **React** |
| --- | --- |
| onclick | onClick |
| onchange | onChange |
| onsubmit | onSubmit |
| onkeydown | onKeyDown |
| onmouseover | onMouseOver |

#### Example:

#### <input type="text" onChange={handleChange} /><button onClick={handleClick}>Submit</button>

**Output for this lab:**

****

****