**11. ReactJS-HOL**

**Objectives:**

**Q1. Explain React events**

React events are the React version of DOM events. They are used to handle user interactions like clicks, form submissions, key presses, etc. React events are wrapped in a cross-browser wrapper called SyntheticEvent to ensure consistency across browsers.

**Q2. Explain about event handlers**

Event handlers are functions in React that respond to user actions. They are passed as props to JSX elements using camelCase syntax.  
Example:

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

Here, handleClick is an event handler function executed when the button is clicked.

**Q3. Define Synthetic event**

A Synthetic Event is a wrapper around the native browser event in React. It combines all the browser differences and provides a consistent API. Synthetic events are part of React’s event delegation system and work identically across all browsers.

**Q4. Identify React event naming convention**

React uses camelCase for event names instead of lowercase like HTML. For example:

* HTML: <button onclick="doSomething()">
* React: <button onClick={doSomething}>

This is consistent across all React events such as:

* onClick
* onChange
* onSubmit
* onMouseEnter
* onKeyDown

**Hands-On exercise:**

**Code:  
App.js**  
import React, { useState } from 'react';

import './App.css';

function App() {

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

  const [amount, setAmount] = useState('');

  const [currency, setCurrency] = useState('');

  const [message, setMessage] = useState('');

const increment = () => {

    setCount(prev => prev + 1);

    sayHello();

  };

const decrement = () => {

    setCount(prev => prev - 1);

  };

const sayHello = () => {

    alert('Hello! Counter incremented');

  };

const sayWelcome = (msg) => {

    alert(msg);

  };

const handlePress = (e) => {

    alert("I was clicked");

  };

const handleSubmit = (e) => {

    e.preventDefault();

    const result = (parseFloat(amount) / 90).toFixed(2);

     alert("Converting to euro amount is " + result);

     setCurrency("Euro")

  };

return (

    <div className="App">

      <h2>Counter: {count}</h2>

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

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

      <button onClick={() => sayWelcome("Welcome")}>Say Welcome</button>{' '}

      <button onClick={handlePress}>Click on me</button>

      <h2 style={{ color: "green" }}>Currency Convertor!!!</h2>

      <form onSubmit={handleSubmit}>

        <label>Amount:</label><br />

        <input

          type="number"

          value={amount}

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

        /><br />

        <label>Currency:</label><br />

        <input type="text" value={currency} readOnly /><br /><br />

        <button type="submit">Submit</button>

      </form>

    </div>

  );

}

export default App;

**App.css**

.App {

  margin: 30px;

  font-family: Arial, sans-serif;

}

.App-logo {

  height: 40vmin;

  pointer-events: none;

}

@media (prefers-reduced-motion: no-preference) {

  .App-logo {

    animation: App-logo-spin infinite 20s linear;

  }

}

.App-header {

  background-color: #282c34;

  min-height: 100vh;

  display: flex;

  flex-direction: column;

  align-items: center;

  justify-content: center;

  font-size: calc(10px + 2vmin);

  color: white;

}

**Output:**













