Theory Assignment

❖ Handling Events in React

Question 1: How are events handled in React compared to vanilla JavaScript? What are synthetic events?

Answer:

- In vanilla JavaScript, we use addEventListener to handle events.
- In React, we use JSX props like onClick, onChange, etc.

➢ Synthetic Events:
React uses Synthetic Events, which are wrapper objects around native browser events.
They work the same way across all browsers (cross-browser compatibility).

Question 2: Common Event Handlers in React.js

Answer:
Here are some common event handlers with examples:

1. onClick – When a button is clicked:

```
<button onClick={() => alert("Button clicked!")}>Click Me</button>
```

2. onChange – When input text changes:

```
<input type="text" onChange={(e) => console.log(e.target.value)} />
```

3. onSubmit – When a form is submitted:

```
<form onSubmit={(e) => { e.preventDefault(); alert("Form submitted"); }}>
  <button type="submit">Submit</button>
</form>
```

Question 3: Why bind event handlers in class components?

Answer:
In class components, you need to bind this to the method because:

- By default, this is undefined in event handlers.
- Binding ensures that this refers to the component instance.

➢ Example without binding (⚠️ will not work):

```
class MyComp extends React.Component {
  handleClick() {
    console.log(this); // undefined
  }
  render() {
    return <button onClick={this.handleClick}>Click</button>;
  }
}
```

Fixed using binding:

```
<button onClick={this.handleClick.bind(this)}>Click</button>
```

Or bind in constructor:

```
constructor() {
  super();
  this.handleClick = this.handleClick.bind(this);
}
```

Or use an arrow function:

```
handleClick = () => {
  console.log(this); // works fine
}
```