Training Day 13

Day 13 - 7th July 2025

TOPICS COVERED – React Basics: State, Virtual DOM, App.jsx

1. state Variable in React

- React components use state to store dynamic data that affects what is rendered.
- Declared using the useState hook (in functional components).

Example:

```
import { useState } from "react";

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

  return (
    <div>
      <p>Count is: {count}</p>
      <button onClick={() => setCount(count + 1)}>Increase</button>
    </div>
  );
}
```

useState(0) initializes the count at 0.

setCount updates the state and re-renders the component.

2. Virtual DOM in React

- A lightweight in-memory representation of the actual DOM.
- React uses it to efficiently update the UI by only rendering changes, not the full page.
- Improves performance and user experience.

React compares old and new virtual DOMs using **diffing** and updates only the changed parts.

3. Introduction to App.jsx

- App.jsx is the root component of a React project (especially when using Vite or CRA).
- Acts like the main container for all other components.

Example Structure:

```
function App() {
  return (
    <div>
      <h1>Welcome to My App</h1>
      <Counter />
    </div>
  );
}

export default App;
```

Use it to import and render other components.

Always export it using export default.

3. Introduction to App.jsx

- App.jsx is the main/root component in React apps.
- It typically holds global layout and renders child components.

Example:

```
jsx
CopyEdit
function App() {
  return (
    <div>
      <h1>My React App</h1>
      <Counter />
    </div>
  );
}
```

export default App;

4. Reconciliation in React

- Reconciliation is React's process of comparing the old virtual DOM with the new one.
- React determines which parts of the DOM actually need to be updated.

Helps in improving efficiency and performance of the UI rendering.

5. Async Nature of useState

- useState updates **asynchronously** — it does not immediately change the value of the variable.
- If you log the value right after calling setState, it may still show the old value.

Example:

jsx
CopyEdit
const [value, setValue] = useState(0);

setValue(1);
console.log(value); // Still 0, not 1

React **batches** updates for performance, and the new value is available **after** the next render cycle.

TASK FOR TOMORROW

- Learn about:
    - **Props** in React
    - Practice creating multiple components and passing data using props