Training Day 16

TOPICS COVERED

- **Uplifting State**

Uplifting state means moving the state up to the closest common ancestor of two or more child components that need to share or affect that state.

Why uplift?

Sometimes sibling components need to communicate, but since React follows one-way data flow (parent → child), we "lift" the state to a common parent.

Example:

```
function Parent() {
  const [message, setMessage] = useState("");
  return (
    <>
      <InputComponent setMessage={setMessage} />
      <DisplayComponent message={message} />
    </>
  );
}

function InputComponent({ setMessage }) {
  return <input onChange={(e) => setMessage(e.target.value)} />;
}

function DisplayComponent({ message }) {
  return <p>Message: {message}</p>;
}
```

InputComponent updates the state, and DisplayComponent reads it — both through the parent.

- **Props Drilling**

Props are used to pass data down from parent to child components. However, when you pass data through many layers unnecessarily, it becomes props drilling.

```
function App() {
  return <Parent user="xyz" />;
}

function Parent({ user }) {
  return <Child user={user} />;
}

function Child({ user }) {
  return <h2>Hello, {user}!</h2>;
}
```

- **Context API**

The Context API provides a way to share data globally without props drilling.

Useful for:

Authentication

User info

Theme switching

Language settings

Steps:

```
// Create context
```

```
const UserContext = createContext();

// Provide context
function App() {
  return (
    <UserContext.Provider value="xyz">
      <Child />
    </UserContext.Provider>
  );
}

// Use context
function Child() {
  const user = useContext(UserContext);
  return <h2>Hello, {user}!</h2>;
}
```

TOOLS USED

Visual Studio Code (VS Code)

React Developer Tools Extension

Browser console