Training Day 17

TOPICS COVERED

- **Context API**

The Context API allows data to be shared across components without manually passing props through every level — solving the problem of props drilling.

Steps:

1. Create Context

```
const ThemeContext = createContext();
```

2. Provide Context (typically in App.js)

```
function App() {
  return (
    <ThemeContext.Provider value="dark">
      <Navbar />
    </ThemeContext.Provider> );
}
```

3. Consume Context (in any child component)

```
function Navbar() {
  const theme = useContext(ThemeContext);
  return <div className={theme}>Welcome</div>;
}
```

- **Dynamic Routing (React Router)**

Dynamic Routing allows us to navigate between pages based on URL parameters, enabling more flexible and scalable apps.

Basic Setup:

```
import { BrowserRouter, Routes, Route, useParams } from "react-router-dom";
function App() {
  return (
  <BrowserRouter>
   <Routes>
    <Route path="/" element={<Home />} />
    <Route path="/user/:id" element={<User />} />
   </Routes>
  </BrowserRouter>);}
```

Dynamic Route Component:

```
function User() {
  const { id } = useParams();
  return <h2>User ID: {id}</h2>;
}
```

Visiting /user/123 → renders User ID: 123

- **Introduction to Backend (Node.js + Express)**

What is Backend?

The backend is the server-side part of an application — it handles data storage, logic, authentication, and interaction with databases or external APIs.

Backend Technologies:

Node.js → runtime to run JavaScript on the server

Express.js → a web framework for building APIs and routes

MongoDB (commonly used) → NoSQL database

Features backend handles:

APIs & routing

Database operations

Authentication

File uploads