Day 17 – [11th July 2025]

TOPICS COVERED

Context API in React:

The Context API allows us to share state across components without manually passing props (solves props drilling).

Key Points:

- ➢ Create a context using createContext()
- ➢ Use Provider to wrap components
- ➢ Use useContext() to access shared values

Example:

```
import { createContext, useContext } from "react";
const ThemeContext = createContext();
function App() {
  return (
    <ThemeContext.Provider value="dark">
      <Page />
    </ThemeContext.Provider>
  );
}
function Page() {
  const theme = useContext(ThemeContext);
  return <h1>Current Theme: {theme}</h1>;
}
```

Dynamic Routing in React:

Dynamic routing means using variables (like user ID or product name) inside URLs using react-router-dom.

Installation:

```
npm install react-router-dom
```

Example:

```
import { BrowserRouter, Routes, Route, useParams } from "react-router-dom";
function User() {
  const { id } = useParams();
  return <h2>User ID: {id}</h2>;
}
function App() {
  return (
    <BrowserRouter>
      <Routes>
        <Route path="/user/:id" element={<User />} />
      </Routes>
    </BrowserRouter>
  );
}
```

URL /user/5 will display: User ID: 5

Backend Setup with Express.js:

Express.js is a lightweight Node.js framework used for backend development.

Setup:

- npm init -y
- npm install express

Simple Server Code:

```
const express = require("express");
const app = express();
const PORT = 3000;
app.get("/", (req, res) => {
  res.send("Hello from Express!");
});
app.listen(PORT, () => {
  console.log(`Server running at http://localhost:${PORT}`);
});
```

Run using:

```
node index.js
```

TOOLS USED:

VS Code

Node.js + Express.js

React Router DOM

Browser for testing dynamic routes

Postman/Hoppscotch (to test Express API)