WEB APP DEVELOPMENT WITH REACTJS (INT252)

Lecture 15: Nested Routes & Layout Components

Unit IV – Routing & Advanced Concepts

Syllabus Mapping:

- Nested routing
- Layout components
- Shared UI using React Router

Target Learner: Beginner → Intermediate

Why Nested Routes Are Needed

So far we have:

- Independent pages

Real applications have:

- Common layout (Navbar, Sidebar)
- Pages inside pages

Real App Example

```
/dashboard
/dashboard/profile
/dashboard/settings
```

All share the same layout

What Are Nested Routes?

Nested routes mean:

- Routes inside routes
- Parent + child relationship

Visual Structure

```
Dashboard (Layout)
    ├── Profile
    └── Settings
```

Layout Component Concept

A layout component:

- Contains shared UI
- Renders child routes

Introducing `<Outlet>`

`<Outlet>` is:

- A placeholder
- Where child routes render

Basic Layout Component

```
import { Outlet } from 'react-router-dom';

function DashboardLayout() {
  return (
    <>
      <h2>Dashboard</h2>
      <Outlet />
    </>
  );
}
```

Defining Nested Routes

```
<Routes>
  <Route path="/dashboard" element={<DashboardLayout />}>
    <Route path="profile" element={<Profile />} />
    <Route path="settings" element={<Settings />} />
  </Route>
</Routes>
```

How Routing Works Here

```
/dashboard/profile
    ↓
DashboardLayout renders
    ↓
Profile appears in <Outlet>
```

Navigating Nested Routes

```
<Link to="profile">Profile</Link>
<Link to="settings">Settings</Link>
```

(Relative paths)

Full Example Structure

```
App
 └── DashboardLayout
       ├── Profile
       └── Settings
```

Why Layout Routes Are Powerful

- No repeated navbar code
- Clean routing
- Scalable structure

Common Beginner Mistakes

❌ Forgetting `<Outlet>`
❌ Using absolute paths wrongly
❌ Nesting routes incorrectly

Practice Exercises

1. Create `/admin` layout
2. Add `/admin/users` and `/admin/reports`
3. Render children using `<Outlet>`

✅ Answers – Practice Exercises

```
<Route path="/admin" element={<AdminLayout />}>
  <Route path="users" element={<Users />} />
  <Route path="reports" element={<Reports />} />
</Route>
```

Key Takeaways

- Nested routes create hierarchy
- Layout components share UI
- `<Outlet>` renders child routes

Next Lecture

Lecture 16: Programmatic Navigation & useNavigate

Unit IV – Routing & Advanced Concepts