



Experiment 2

Student Name: Sagen Hansda

UID: 23BCS12396

Branch: CSE

Section/Group: KRG 3-B

Semester: 6th

Date of Performance: 20/01/2026

Subject Name: Full Stack Development – II

Subject Code: 23CSH-309

1. Aim: To implement Single Page Application (SPA) routing in the EcoTrack application using React Router, secure application routes using protected routing with Context API-based authentication, and manage shared authentication state across components.

2. Objective:

- To configure client-side routing using React Router
- To implement SPA navigation without page reloads
- To protect routes using authentication-based route guards
- To manage shared authentication state using React Context API
- To implement login and logout functionality
- To restrict unauthorized access to protected pages
- To understand redirection logic in protected routing
- To analyze the role of Context API in state management

3. Implementation / Code:

Tools & Technologies Used:

- AWS Free Tier Account
- Web Browser (Google Chrome / Firefox)
- Amazon EC2 Service
- RDP Client (Microsoft Remote Desktop)
- Internet-enabled Laptop/Desktop

Implementation Description:

- The EcoTrack application is enhanced by implementing client-side routing using React Router, enabling seamless navigation between different pages without full page reloads.
- An authentication system is implemented using React Context API, which stores and manages the authentication state (`isAuthenticated`) across the entire application.
- A `ProtectedRoute` component is created to restrict access to sensitive pages such as Dashboard, Logs, and Data. If the user is not authenticated, they are automatically redirected to the Login page.
- Login functionality updates the authentication state using context, while logout functionality resets the authentication state and redirects the user back to the login page.



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

- This approach ensures secure navigation, centralized state management, and a smooth SPA user experience.

Sample Code Snippet:

```
⚙️ Header.jsx ×  
src > components > ⚙️ Header.jsx > ...  
1 import { Link } from "react-router-dom";  
2 import { useAuth } from "../context/AuthContext";  
3  
4 const Header = () => {  
5   const { isAuthenticated, logout } = useAuth();  
6  
7   return (  
8     <header style={{ background: "#6BCF8E", padding: "15px" }}>  
9       <h2>EcoTrack</h2>  
10      <nav style={{ display: "flex", gap: "15px" }}>  
11        <Link to="/">Dashboard</Link>  
12  
13        {isAuthenticated && (  
14          <>  
15            <Link to="/logs">Logs</Link>  
16            <Link to="/data">Data</Link>  
17          </>  
18        )}  
19  
20        {!isAuthenticated && <Link to="/login">Login</Link>}  
21        {isAuthenticated && <button onClick={logout}>Logout</button>}  
22      </nav>  
23    </header>  
24  );  
25};  
26  
27  
28 export default Header;  
29
```

```
⚙️ Dashboard.jsx ×  
src > pages > ⚙️ Dashboard.jsx > ...  
1 const Dashboard = () => {  
2   return (  
3     <div style={{ padding: "20px" }}>  
4       <h2>Dashboard</h2>  
5       <p>Welcome to EcoTrack Dashboard</p>  
6     </div>  
7   );  
8 };  
9  
10 export default Dashboard;  
11
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

The screenshot shows a code editor interface with a dark theme. On the left is the 'EXPLORER' sidebar, which lists the project structure:

- OPEN EDITORS
- AuthContext.jsx (selected)
- ECOTRACK
- node_modules
- public
- src
 - assets
 - components
 - Header.jsx
 - context
 - AuthContext.jsx
 - pages
 - Dashboard.jsx
 - Data.jsx
 - Login.jsx
 - Logs.jsx
 - routes
 - ProtectedRoute.jsx
- # App.css
- # App.jsx
- # index.css
- # main.jsx
- .gitignore

```
AuthContext.jsx ×

src > context > AuthContext.jsx > ...
1 import { createContext, useContext, useState } from "react";
2
3 const AuthContext = createContext(null);
4
5 export const AuthProvider = ({ children }) => {
6   const [isAuthenticated, setIsAuthenticated] = useState(false);
7
8   const login = () => setIsAuthenticated(true);
9   const logout = () => setIsAuthenticated(false);
10
11   return (
12     <AuthContext.Provider value={{ isAuthenticated, login, logout }}>
13       {children}
14     </AuthContext.Provider>
15   );
16 }
17
18 export const useAuth = () => useContext(AuthContext);
```

The screenshot shows a code editor interface with a dark theme. On the left is the 'EXPLORER' sidebar, which lists the project structure:

- src > pages > Login.jsx > ...
- AuthContext.jsx
- Dashboard.jsx
- Data.jsx
- Login.jsx (selected)
- Logs.jsx
- ProtectedRoute.jsx
- # App.css
- # App.jsx
- # index.css
- # main.jsx
- .gitignore

```
Login.jsx ×

src > pages > Login.jsx > ...
1 import { useNavigate } from "react-router-dom";
2 import { useAuth } from "../context/AuthContext";
3
4 const Login = () => {
5   const { login } = useAuth();
6   const navigate = useNavigate();
7
8   const handleLogin = () => {
9     login();
10    navigate("/");
11  };
12
13  return (
14    <div style={{ padding: "20px" }}>
15      <h2>Login</h2>
16      <button onClick={handleLogin}>Login to EcoTrack</button>
17    </div>
18  );
19}
20
21 export default Login;
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CHANDIGARH
UNIVERSITY

Discover. Learn. Empower.

ProtectedRoute.jsx

```
src > routes > ProtectedRoute.jsx > ...
1 import { Navigate } from "react-router-dom";
2 import { useAuth } from "../context/AuthContext";
3
4 const ProtectedRoute = ({ children }) => {
5   const { isAuthenticated } = useAuth();
6
7   return isAuthenticated ? children : <Navigate to="/login" replace />;
8 }
9
10 export default ProtectedRoute;
11
```

4. Output:

- The EcoTrack application successfully implements SPA routing
- Navigation occurs without full page reloads
- Unauthorized users are redirected to the login page
- Authenticated users can access Dashboard, Logs, and Data pages
- System logs and environmental data are displayed dynamically
- Logout functionality securely ends the session
- Proper route protection is verified using ProtectedRoute

EcoTrack

[Dashboard](#) [Logs](#) [Login](#)

[Summary](#) | [Analytics](#)

Environmental Data

ID	Category	Value	Impact Level
1	Electricity Usage	120 kWh	Medium
2	Water Consumption	450 Liters	Low
3	Carbon Emission	18 kg CO ₂	High
4	Waste Generated	6 kg	Medium





DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CHANDIGARH
UNIVERSITY

Discover. Learn. Empower.

5. Learning Outcomes (What I Have Learnt)

After completing this experiment, the student is able to:

- Implement SPA routing using React Router
- Secure application routes using protected routing
- Manage shared authentication state using Context API
- Implement login and logout functionality
- Understand route redirection logic
- Compare Context API with Redux at an introductory level
- Build scalable and secure React applications