



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Experiment 2

Student Name: Myank

Branch: BE CSE

Semester: 6th

Subject Name: Full Stack - II

UID: 23BCS10894

Section/Group: KRG_3B

Date of Performance: 19/01/26

Subject Code: 23CSH-309

Aim: The aim of this implementation is to develop a **secure navigation system** using **Context-based authentication along with route protection**, ensuring that only logged-in users can access authorized pages of the application.

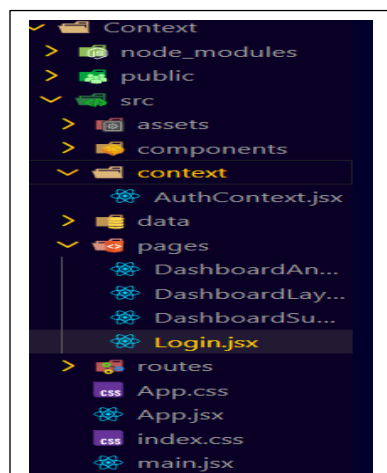
Objective:

- To manage **user authentication state globally** using Context
- To implement **routing for page navigation**
- To restrict access to protected routes for unauthenticated users
- To allow page switching **only after successful login**
- To redirect unauthenticated users automatically to the **login page**
- To improve application security and user experience
- To maintain clean, scalable, and reusable code structure

Input/Apparatus Used:

- Programming Language: JavaScript (ES6+)
- Framework / Library: React (Functional Components)
- Build Tool: Vite
- Code Editor: Visual Studio Code
- Web Browser: Google Chrome

Files Structure





DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Flow of Data

App Load



Check Login Status (Context)



Is User Logged In?



NO → Show Login Page



YES → Allow Access to Protected Pages

Steps

1. Application Starts

- index.js renders `<App />`
- `<App />` is wrapped with `AuthProvider`

2. AuthContext Controls Login State

AuthContext.js

- Stores:
 - `isLoggedIn`
 - `login()`
 - `logout()`

3. Routes Are Defined in App.jsx

Routes:

`"/login"` → Login Page

`"/dashboard"` → Protected

`"/profile"` → Protected

4. ProtectedRoute Acts as a Guard

ProtectedRoute.js logic:

IF `isLoggedIn === true`

→ Render Requested Page

ELSE

→ Redirect to Login Page

5. User Accesses the App (Not Logged In)

Flow:

User enters `/dashboard`



ProtectedRoute checks `isLoggedIn`

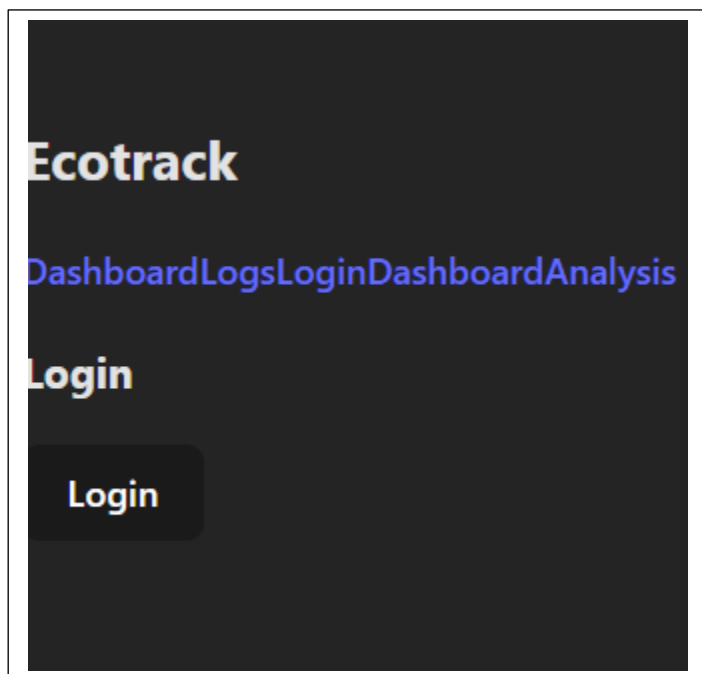


FALSE



Redirect → `/login`

Output



Learning Outcomes

- Learned global state management using Context
- Implemented protected routing for secure navigation
- Controlled access based on authentication status
- Improved security and code maintainability