DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

Experiment 3

Student Name: Madhav
UID: 23BAI70107
Branch: BE CSE AIML
Section/Group: 23AIT-KRG G1
Semester: 6th
Date of Performance: 28 January 2026
Subject Name: Full Stack
Subject Code: 23CSH-382

1. **Title:**
   Redux Toolkit & Asynchronous State Management in EcoTrack

2. **Aim:**
   To implement centralized state management in the EcoTrack application using Redux Toolkit and to handle asynchronous data operations using Redux async thunks with proper loading and error states.

3. **Objective:**
   After completing this experiment and its follow-up tasks, the student will be able to:
   1. Configure a Redux store in a React application using Redux Toolkit
   2. Create and integrate Redux slices for managing application data
   3. Implement asynchronous actions using Redux async thunks
   4. Manage loading, success, and error states during asynchronous operations
   5. Connect React components to Redux state using React-Redux hooks
   6. Trigger asynchronous data fetching through Redux actions from UI components
   7. Use Redux state to derive filtered views without modifying the global store
   8. Enhance user experience by handling refresh actions and improving async UI feedback.

4. Implementation/Code:

- logSlice.jsx:

```jsx
import { createSlice, createAsyncThunk} from "@reduxjs/toolkit";

export const fetchLogs = createAsyncThunk(
    "logs/fetchLogs",
    async() => {
        await new Promise((resolve) => setTimeout(resolve, 1000));

        return [
            { id: 1, activity: "Car Travel", carbon: 4 },
            { id: 2, activity: "Electricity Usage", carbon: 6 },
            { id: 3, activity: "Cycling", carbon: 0 },
        ]
    }
)

const logsSlice = createSlice({
    name : "logs",
    initialState : {
        data : [],
        status : "idle",
        error : null,
    },
    reducers : {},
    extraReducers : (builder) => {
        builder
        .addCase(fetchLogs.pending, (state, action) =>{
            state.status = "loading";
        })
        .addCase(fetchLogs.fulfilled, (state, action) =>{
            state.status = "success";
            state.data = action.payload;
        })
        .addCase(fetchLogs.rejected, (state, action) =>{
            state.status = "failed";
            state.error = action.error.message;
        })
    }
})

export default logsSlice.reducer;
```

- Logs.jsx:

```jsx
import { useEffect } from "react";
import { useDispatch, useSelector } from "react-redux";
import { fetchLogs } from "../store/logSlice";


const Logs = () => {
    const dispatch = useDispatch();
    const {data, status, error} = useSelector((state) => state.logs);

    const handleRefresh = () => {
        dispatch(fetchLogs());
    }

    useEffect(() => {
        if(status === "idle") {
            dispatch(fetchLogs());
        }
    }, [status, dispatch]);
    if(status === "loading") {
        return <p>Loading Logs...</p>
    }
    if(status === "failed") {
        return <p>Error : {error}</p>
    }

    return (
        <div>
            <h1>Logs</h1>
            <ol>
                {data.map((log) => (
                    <li key={log.id}>
                        {log.activity}: {log.carbon} kg CO2
                    </li>
                ))}
            </ol>
            <button onClick={handleRefresh}>Refresh</button>
        </div>
    )
}

export default Logs;
```

- Store.jsx:

```jsx
import { configureStore } from "@reduxjs/toolkit";
import logsReducer from "./logSlice"

const store = configureStore({
    reducer : {
        logs : logsReducer,
    },
});

export default store;
```

- AuthContext.jsx:

```jsx
import { createContext, useContext, useState } from "react";

const AuthContext = createContext(null);

export const AuthProvider = ({children}) => {
    const [isAuthenticated, setIsAuthenticated] = useState(false);

    return (
        <AuthContext.Provider value = {{isAuthenticated, setIsAuthenticated}}>
            {children}
        </AuthContext.Provider>
    )
}

export const useAuth = () => useContext(AuthContext);
```

- ProtectedRoute.jsx:

```jsx
import { Navigate } from "react-router-dom";
import { useAuth } from "../context/AuthContext";
import { children } from "react";

const ProtectedRoute = ({children}) => {
    const {isAuthenticated} = useAuth();

    if(!isAuthenticated) {
        return <Navigate to = "/login" replace/>
    }
    return children;
}

export default ProtectedRoute;
```

- Login.jsx:

```jsx
import { useAuth } from "../context/AuthContext";
import { useNavigate } from "react-router-dom";

const Login = () => {
    const {setIsAuthenticated} = useAuth();
    const navigate = useNavigate();

    const handleLogin = () => {
        setIsAuthenticated(true);
        navigate("/");
    }
    return (
        <>
        <h3>Login</h3>
        <button onClick={handleLogin}>Login</button>
        </>
    )
}
export default Login;
```

- Logout.jsx:

```jsx
import { useEffect } from "react";
import { useNavigate } from "react-router-dom";
import { useAuth } from "../context/AuthContext";

const Logout = () => {
  const { setIsAuthenticated } = useAuth();
  const navigate = useNavigate();

  useEffect(() => {
    setIsAuthenticated(false);
    navigate("/login");
  }, [setIsAuthenticated, navigate]);

  return <p>Logging you out...</p>;
}

export default Logout;
```

- DashboardSettings, DashboardSummary, DashboardAnalytics:

```jsx
const DashboardSettings = () => {
    return (
        <h3>These are the settings</h3>
    )
}

export default DashboardSettings;
```

```jsx
const DashboardSummary = () => {
    return (
        <h3>This is a Summary</h3>
    )
}

export default DashboardSummary;
```

```jsx
const DashboardAnalytics = () => {
    return (
        <h3>This is a Analysis</h3>
    )
}

export default DashboardAnalytics;
```

- DashboardLayout.jsx:

```jsx
import { Link, Outlet } from "react-router-dom";

const DashboardLayout = () => {
    return (
        <>
        <h3>Dashboard</h3>

        <nav>
            <Link to = "settings">Settings</Link>{" "}
            <Link to = "summary">Summmary</Link>{" "}
            <Link to = "analytics">Analytics</Link>
        </nav>

        <hr />
        <Outlet />
        </>

    )
}

export default DashboardLayout;
```

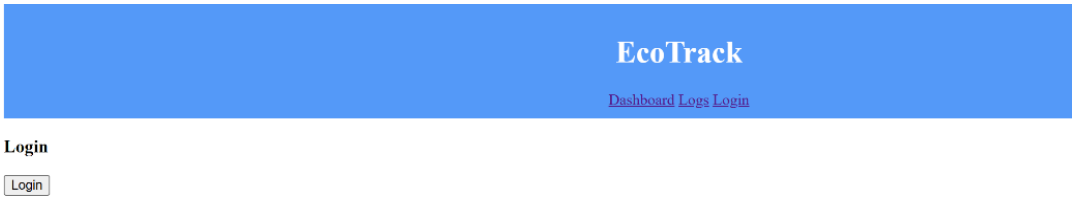
- Header.jsx:

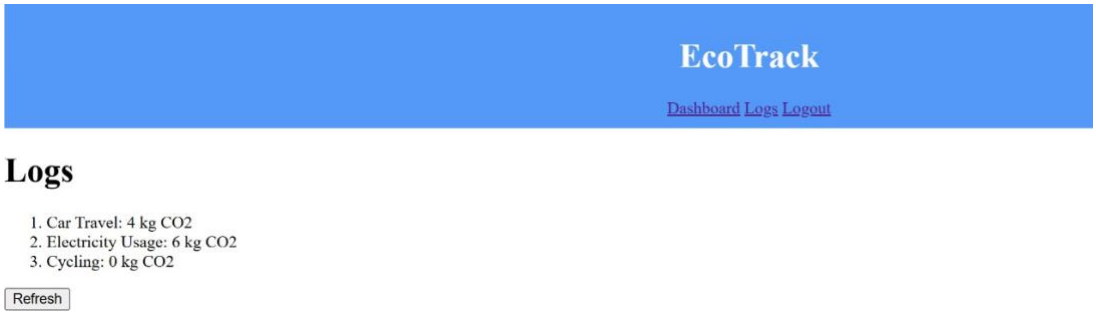
```jsx
import { Link } from "react-router-dom";
import { useAuth } from "../context/AuthContext";

const Header = () => {
    const {isAuthenticated} = useAuth();
    return (
        <header style = {{
            padding: '10px',
            backgroundColor: '#5499f8',
            color : 'white',
            textAlign: 'center',
        }}>
            <h1>EcoTrack</h1>
            <Link to = "/">Dashboard</Link>{" "}
            <Link to = "/logs">Logs</Link>{" "}
            {isAuthenticated ? (
                <Link to = "/logout">Logout</Link>
            ) : (
                <Link to = "/login">Login</Link>
            )}
        </header>
    )
}
export default Header;
```

- App.jsx:

```jsx
import { Route, Routes } from "react-router-dom";
import Login from "./pages/Login";
import Logout from "./pages/Logout";
import DashboardAnalytics from "./pages/DashboardAnalytics";
import DashboardLayout from "./pages/DashboardLayout";
import DashboardSummary from "./pages/DashboardSummary";
import DashboardSettings from "./pages/DashboardSettings";
import ProtectedRoute from "./routes/ProtectedRoute";
import Logs from "./pages/logs";
import Header from "./components/Header";

function App() {
    return (
        <>
        <Header />
        <Routes>
            <Route path = "/Login" element = {<Login/>} />
            <Route path = "/Logout" element = {<Logout/>}/>
            <Route path = "/"
            element = {
                <ProtectedRoute>
                <DashboardLayout/>
                </ProtectedRoute>
            }>
            <Route index element = {<DashboardSummary/>}/>
            <Route path = "settings" element = {<DashboardSettings/>}/>
            <Route path = "summary" element = {<DashboardSummary/>}/>
            <Route path = "analytics" element = {<DashboardAnalytics/>}/>
            </Route>
            <Route path = "/logs"
            element = {
                <ProtectedRoute>
                <Logs/>
                </ProtectedRoute>
            }>
            </Route>
        </Routes>
        </>
    )
}
export default App;
```

5. Output





6. Learning Outcome

- We learnt about React Apps and how to create them.
- We learnt about redux and it's components.
- We learnt about the use of thunks andSlices.
- We learnt about Authentication and index pages
- We learnt the use of useContext and useState.
- We learnt about the flow of a React project.