



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Experiment 3

Student Name: Omprakash
Branch: BE- CSE(AIML)
Semester: 6th
Subject Name: Full Stack

UID: 23BAI70296
Section: 23AIT KRG 1(G2)
Date of Performance: 29/01/26
Subject Code: 23CSH-382

- 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.

2. Objective:

After completing this experiment and its follow-up task, the student will be able to:

- Configure a Redux store in a React application using Redux Toolkit
- Create and integrate Redux slices for managing application data
- Implement asynchronous actions using Redux async thunks
- Manage loading, success, and error states during asynchronous operations
- Connect React components to Redux state using React-Redux hooks
- Trigger asynchronous data fetching through Redux actions from UI components
- Use Redux state to derive filtered views without modifying the global store
- Enhance user experience by handling refresh actions and improving async UI feedback.

3. Implementation/Code:

Header.js:

```
import Link } from "react-router-dom";  
  
const Header = () => {  
  
  return (
```

```

<header style = {{{
  padding: '10px',
  backgroundColor: "rgb(224, 168, 107)",
  color : 'white',
  textAlign: 'center',
}}}>

<h1>EcoTrack-2</h1>

<Link style={{paddingRight: "10px", color: "#00ffff0" }} to = "/">Dashboard</Link>
<Link style={{padding: "10px", color: "#00ffff0" }} to = "/logs">Logs</Link>
<Link style={{padding: "10px", color: "#00ffff0" }} to = "/login">Login</Link>

</header>

)

}

export default Header;

```

AuthContext.js:

```

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);
```

logs.js:

```
export const logs = [
```

```

{ id: 1, activity: "Car Travel", carbon: 4 },
{ id: 2, activity: "Electricity Usage", carbon: 6 },
{ id: 3, activity: "Cycling", carbon: 0 },
{ id: 4, activity: "Bus Travel", carbon: 3 },
{ id: 5, activity: "Solar Energy Usage", carbon: 1 },
{ id: 6, activity: "Flight Travel", carbon: 8 },
];
export default logs;

```

Dashboard.js:

```

import logs from './data/logs'
import Header from './components/header'
const totalCarbon = logs.reduce ((sum, log) => sum+log.carbon,0)

const Dashboard = () => {
  return (
    <div>
      <h2>{Header}</h2>
      <p style={{padding
        :"1rem",backgroundColor:"#3175a2",color:"#d6eaf8",textAlign:"cente
        r" }}>Total Carbon Footprint: {totalCarbon}</p>

      <ul style={{padding
        :"1rem",backgroundColor:"white",color:"#111",textAlign:"center" }}>

```

```

        {logs.map((log)=>(
            <li key ={log.id}>
                {log.activity}:{log.carbon} kgs
            </li>
        )))
    </ul>
</div>

)
}

export default Dashboard;

```

DashboardAnalytics.js:

```

const DashboardAnalytics =()=>{
    return (
        <p>This is Dashboard Analytics</p>
    )
};

export default DashboardAnalytics;

```

DashboardSummary.js:

```

const DashboardSummary =()=>{
    return (<p>This is Dashboard Summary</p>)
}

export default DashboardSummary;

```

DashboardLayout.js:

```

import { Link,Outlet } from "react-router-dom";
const DashboardLayout = ()=>{

    return (
        <div style={{padding:"1rem"}}>
            <h3>Dashboard</h3>

            <nav>
                <Link to ="/summary">Summary</Link> |{" "}
                <Link to ="/analytics">Analytics</Link>
            </nav>
    
```

```
<hr/>
<Outlet/>

</div>

)
};

export default DashboardLayout;
```

Login.js:

```
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 (
    <div style={{ padding: "1rem" }}>
      <h3>Login Page</h3>
      <button onClick={handleLogin}>Login</button>
    </div>
  )
}

export default Login;
```

Logout.js:

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

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

```

const handleLogout=() => {
    setIsAuthenticated(false);
    navigate("/");
};

return (
    <div style={{padding:"1rem"}}>
        <h3>Logout Page</h3>
        <button onClick={handleLogout}>Logout</button>
    </div>
)
}
}

export default Logout;

```

ViewLogs.js:

```

import logs from "../data/logs";

const Viewlogs = () => {
    return (
        <div>
            <h2>Experiment Logs</h2>
            <ul>
                {logs.map((log, index) => (
                    <li key={log.id}>
                        {log.activity}: {log.carbon} kg CO2
                    </li>
                ))}
            </ul>
        </div>
    )
}

```

export default Viewlogs;

ProtectedRoute.js:

```

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

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

```

```

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

```

LogSlice.js:

```

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:2},
{ 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) => {
state.status = 'loading';
})
.addCase(fetchLogs.fulfilled, (state, action) => {
state.status = 'succeeded';
state.data = action.payload;
})
.addCase(fetchLogs.rejected, (state, action) => {
state.status = 'failed';
state.error = action.error.message;
})
}
});
export default logsSlice.reducer;
;
```

Store.js:

```
import {configureStore} from "@reduxjs/toolkit"
import logsReducer from "./logsSlice";
const store = configureStore({
  reducer :{
    logs: logsReducer,
  },
});
export default store;
```

App.js:

```
import { BrowserRouter as Router, Routes, Route } from "react-router-dom";

import Header from "./components/Header";
import Login from "./pages/login";
import Logs from "./pages/Logs";

import DashboardLayout from "./pages/DashboardLayout";
import DashboardSummary from "./pages/DashboardSummary";
import DashboardSettings from "./pages/DashboardSettings";
import DashboardAnalytics from "./pages/DashboardAnalytics";

import ProtectedRoute from "./routes/ProtectedRoute";

function App() {
  return (
    <Router>
      <Header />

      <Routes>
        <Route path="/login" element={<Login />} />

        <Route
          path="/"
          element={
            <ProtectedRoute>
              <DashboardLayout />
            </ProtectedRoute>
          }
        >
          <Route index element={<DashboardSummary />} />
          <Route path="summary" element={<DashboardSummary />} />
          <Route path="settings" element={<DashboardSettings />} />
          <Route path="analytics" element={<DashboardAnalytics />} />
        </Route>
      <Route
```

```
path="/logs"
element={
<ProtectedRoute>
<Logs />
</ProtectedRoute>
}
/>
</Routes>
</Router>
);
}
```

export default App;

4. Output:



Daily Logs (Redux)

- car travel | 4 kg CO₂
- Electricity Usage | 2 kg CO₂
- Cycling | 0 kg CO₂

5. Learning Outcome

1. Learnt about react files.
2. Learnt Design and apply redux and its uses.
3. Learnt Implement nested routing to build dashboard-style layouts.