



## Experiment 3

**Student Name:** Anuj Yadav

**UID:** 23BIA50011

**Branch:** BE-AIT-CSE

**Section/Group:** 23AML\_KRG-G2

**Semester:** 6<sup>th</sup>

**Date of Performance:** 1 Feb 2026

**Subject Name:** Full Stack II

**Subject Code:** 23CSH-382

### 1. 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:

DashboardAnalytics.jsx:

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

### Header.jsx:

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

const Header = () => {
  return (
    <header style = {{
      padding: '10px',
      backgroundColor: '#3ef381',
      color : 'white',
      textAlign: 'center',
    }}>
      <h1>EcoTrack</h1>
      <Link to = "/">Dashboard</Link>
      <Link to = "/logs">Logs</Link>
      <Link to = "/login">Login</Link>
    </header>
  )
}

export default Header;
```

### Logs.js:

```
const logs = [
  { id: 1, activity: "Car Travel", carbon: 4 },
  { id: 2, activity: "Electricity Usage", carbon: 6 },
  { id: 3, activity: "Cycling", carbon: 0 },
];

export default logs;
```

### DashboardSummary.jsx:

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

export default DashboardSummary;
```

### AuthContext.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);
```

### DashboardLayout.jsx:

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

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

      <nav>
        <Link to = "summary">Summary</Link>
        <Link to = "analytics">Analytics</Link>
      </nav>

      <hr />
      <Outlet />
    </>
  )
}

export default DashboardLayout;
```

### ProtectedRoute.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;
```

### Logout.jsx:

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

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

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

  return (
    <div>
      <button onClick={handleLogout}>Logout</button>
    </div>
  );
};

export default Logout;
```

## Logs.jsx:

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

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

  useEffect(() => {
    if (status === "idle") {
      dispatch(fetchLogs());
    }
  }, [status, dispatch]);

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

  if (status === "loading") {
    return <p>Loading Logs...</p>;
  }

  if (status === "failed") {
    return <p>Error: {error}</p>;
  }

  return (
    <div style={{ padding: "1rem" }}>
      <h3>Daily Logs (Redux)</h3>

      <ul>
        {data.map((log) => (
          <li key={log.id}>
            {log.activity} - {log.carbon} kg CO2
          </li>
        ))}
      </ul>
    </div>
  )
}
```

```

    <button
      onClick={handleRefresh}
      style={{
        marginBottom: "1rem",
        padding: "0.5rem 1rem",
        backgroundColor: "■ #2ecc71",
        color: "■ #fff",
        border: "none",
        borderRadius: "6px",
        cursor: "pointer",
        fontWeight: "600",
        transition: "background-color 0.2s ease, transform 0.1s ease"
      }}
      onMouseOver={(e) => (e.target.style.backgroundColor = "■ #27ae60")}
      onMouseOut={(e) => (e.target.style.backgroundColor = "■ #2ecc71")}
    >
      Refresh
    </button>
  </div>
);
};

export default Logs;

```

### DashboardSettings.jsx:

```

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

export default DashboardSettings;

```



## App.jsx:

```
import { Route, Routes } from "react-router-dom";
import Login from "../pages/Login";
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 = "/"
          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>
    </>
  )
}
```

## Login.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 (
    <div>
      <hr />
      <h2>Login</h2>
      <button onClick={handleLogin}>Login</button>
    </div>
  );
};

export default Login;
```



#### 4. Output:

# EcoTrack-2

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

Loading Logs...

# EcoTrack-2

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

## Daily Logs (Redux)

- Car Travel — 4 kg CO<sub>2</sub>
- Electricity Usage — 6 kg CO<sub>2</sub>
- Cycling — 0 kg CO<sub>2</sub>
- Public Transport — 12 kg CO<sub>2</sub>
- Meat Consumption — 5 kg CO<sub>2</sub>
- Plant-based Meal — 2 kg CO<sub>2</sub>
- Air Travel — 1 kg CO<sub>2</sub>

Refresh

## **5. Learning Outcome:**

- **Configure and integrate a Redux store in a React application using Redux Toolkit.**
- **Create and manage Redux slices for centralized application state handling.**
- **Implement asynchronous operations using Redux async thunks.**
- **Handle loading, success, and error states during asynchronous data fetching.**
- **Connect React components to Redux state using React-Redux hooks.**
- **Improve user experience by managing refresh actions and responsive async UI feedback.**