



### Experiment 3

**Student Name:** Gourav Sharma

**Branch:** BE CSE

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

**Subject Name:** Full Stack Development-II

**UID:** 23BCS10857

**Section/Group:** 23BCSKRG\_3A

**Date of Performance:** 27/01/26

**Subject Code:** 23CSH-309

#### 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:

- To implement centralized state management in EcoTrack using Redux Toolkit.
- To manage asynchronous API calls using Redux async thunks.
- To handle loading, success, and error states effectively in the application.
- To structure Redux slices for scalable frontend architecture.
- To improve data flow and component communication through a global store.

#### 3. Implementation/Code:

##### logsSlice.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: 6 },
      { id: 3, activity: "Cycling", carbon: 0 },
    ];
  }
);
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
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 "./logSlice.js";
```

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

```
export default store;
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Main.jsx:

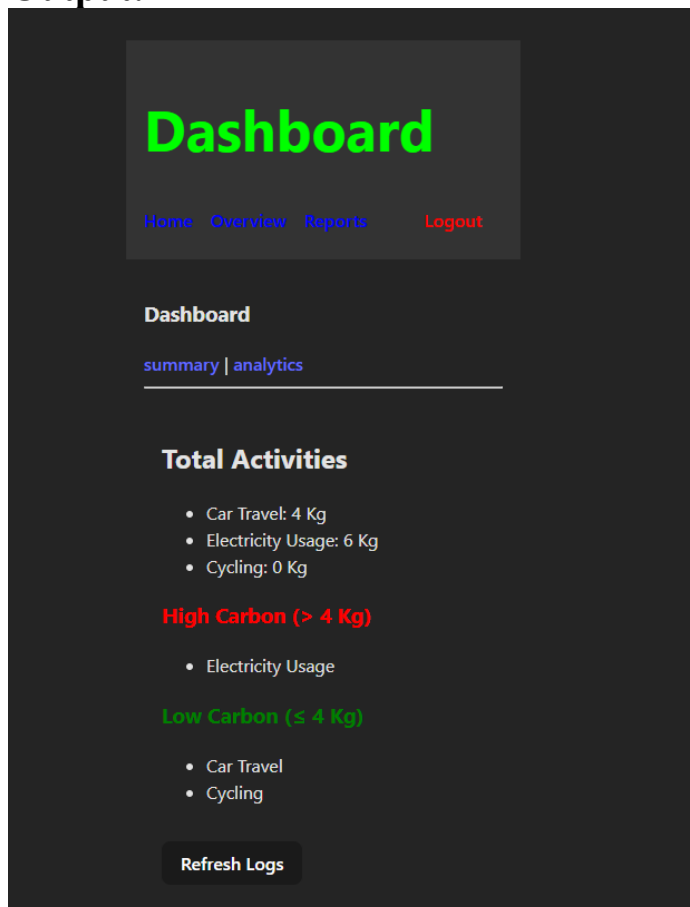
```
import { StrictMode } from "react";
import { createRoot } from "react-dom/client";
import "./index.css";
import App from "./App.jsx";

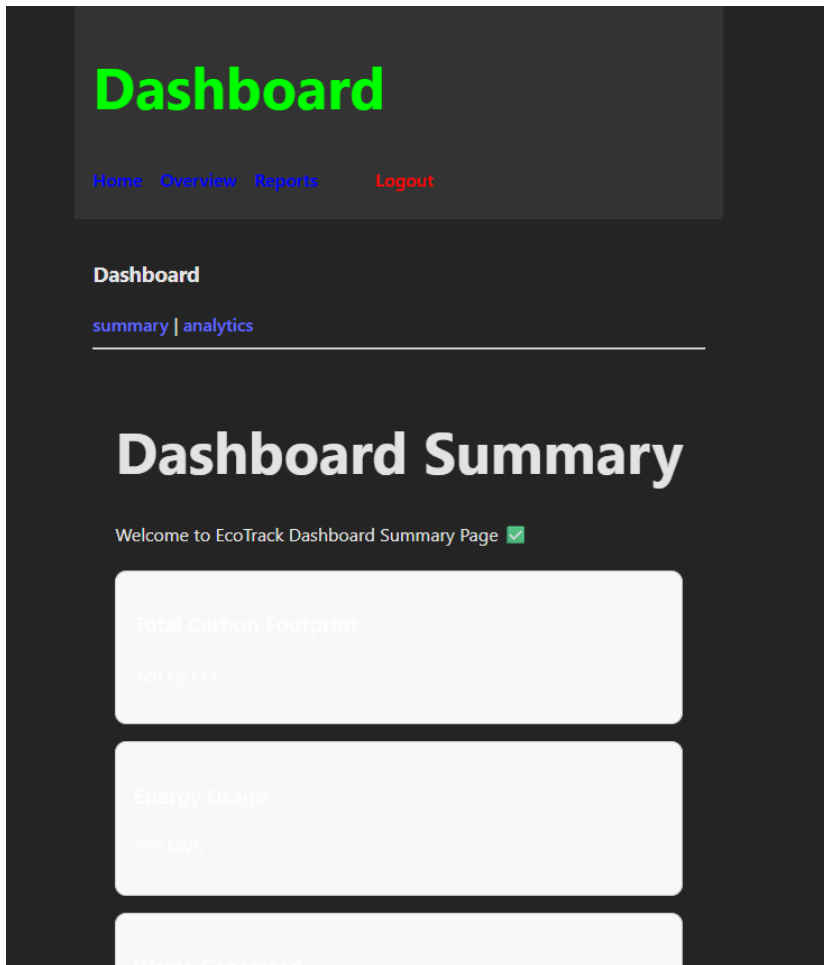
import { Provider } from "react-redux";
import store from "./store/store.js";

import { AuthProvider } from "./context/AuthContext.jsx";

createRoot(document.getElementById("root")).render(
  <StrictMode>
    <Provider store={store}>
      <AuthProvider>
        <App />
      </AuthProvider>
    </Provider>
  </StrictMode>
);
```

## 4. Output:





## 5. Learning Outcome:

- Students will be able to configure Redux Toolkit store and slices.
- Students will understand how to use async thunks for API integration.
- Students will manage loading and error states in real-world React apps.
- Students will design clean and maintainable centralized state logic.
- Students will build production-ready data handling workflows in EcoTrack.