



# CHANDIGARH UNIVERSITY

Discover. Learn. Empower.

## Experiment 3

**Name:** Raj Gupta

**UID:** 23BAI70387

**Branch:** BECE-AIML

**Section/Group:** 23AIT\_KRG-2

**Semester:** 6th

**Date of Performance:** 28/01/2026

**Subject Name:** FULLStack2

**Subject Code:** 23CSH-382

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

2. **Objective:**

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

3. **Implementation/Code:**

a. **logsSlice.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 logSlice = 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 = "success";
        state.data = action.payload;
      })
      .addCase(fetchLogs.rejected, (state, action) => {
        state.status = "failed";
        state.error = action.error.message;
      })
  }
})

export default logSlice.reducer;

```

## b. Slice.jsx

```

import React from 'react'
import {configureStore, createSlice} from '@reduxjs/toolkit';
import LogsReducer from './logsSlice';

```

```
const store = configureStore({
  reducer: {
    logs: LogsReducer,
  },
});
export default store;
```

### c. LogsPage.jsx

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

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

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

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

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

  const logs = data || [];

  return (
    <div>
      {/* Fetch Button */}
      <div style={{ textAlign: "center", marginTop: 20 }}>
        <button
          onClick={handleFetchLogs}
          style={{
            padding: "10px 20px",
```



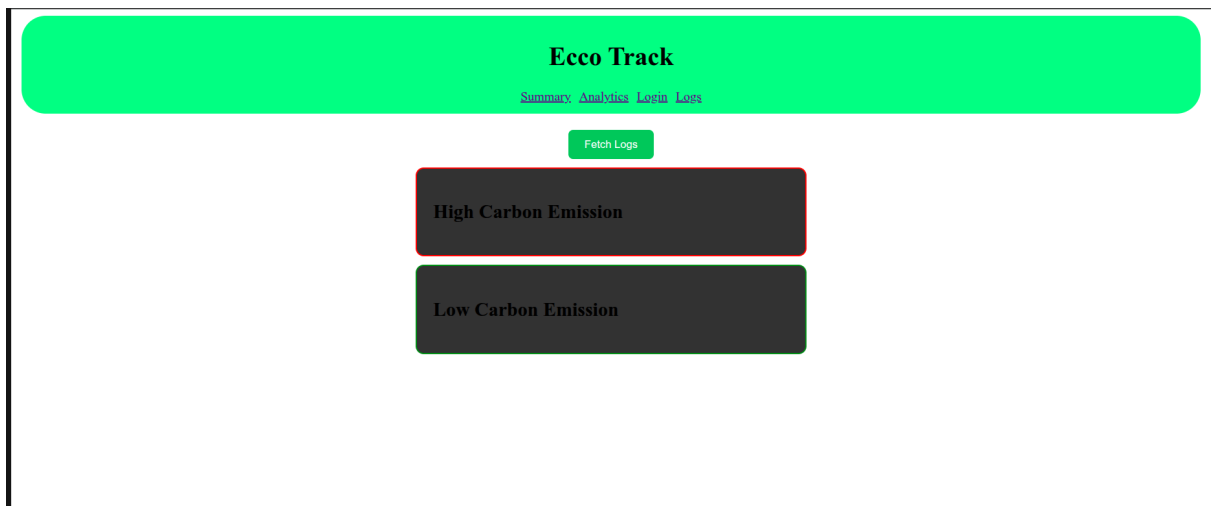
```

        borderRadius: 10,
        border: "2px solid green"
      })>
      <h2>Low Carbon Emission</h2>
      {logs
        .filter(ele => ele.carbon < 4)
        .map((ele, index) => (
          <h3 key={index} style={{ color: "green"
            {ele.activity} {ele.carbon}
          </h3>
        )))
    </div>
  </div>
);
};

export default Logs;

```

#### 4. Output:



Ecco Track

[Summary](#) [Analytics](#) [Login](#) [Logs](#)

Loading your logs.....

Ecco Track

[Summary](#) [Analytics](#) [Login](#) [Logs](#)

Fetch Logs

High Carbon Emission

Car Travel 4

Electricity Usage 6

Low Carbon Emission

Cycling 8

Ecco Track

[Summary](#) [Analytics](#) [Login](#) [Logs](#)

Login

Login Now

Ecco Track

[Summary](#) [Analytics](#) [Login](#) [Logout](#)

Dashboard\_layout

DashboardSummary

Ecco Track

[Summary](#) [Analytics](#) [Login](#) [Logout](#)

Dashboard\_layout

DashboardAnalytics

## Ecco Track

[Summary](#) [Analytics](#) [Login](#) [Logout](#)

**Log Out**

Logout Now

### 5. Learning Outcome:

- a. I learnt how to make redux react.
- b. How the react redux and redux slices actually work.
- c. I learnt how to use the outlet tags along with nested routes.