Experiment 3

Student Name: Karan **UID:**23BAI70265

Branch: BE-AIT-CSE **Section/Group:**23AIT_KRG 1A

Semester:6th **Date of Performance: 29**,Jan 2026

Subject Name: Full Stack - II **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.

Objectives

After completing this experiment and its follow-up task, 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 Code: **Store.js**

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

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

export default store;
```

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: 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, action) => {
            state.status = "pending";
        })
        .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;
        })
    }
}
```

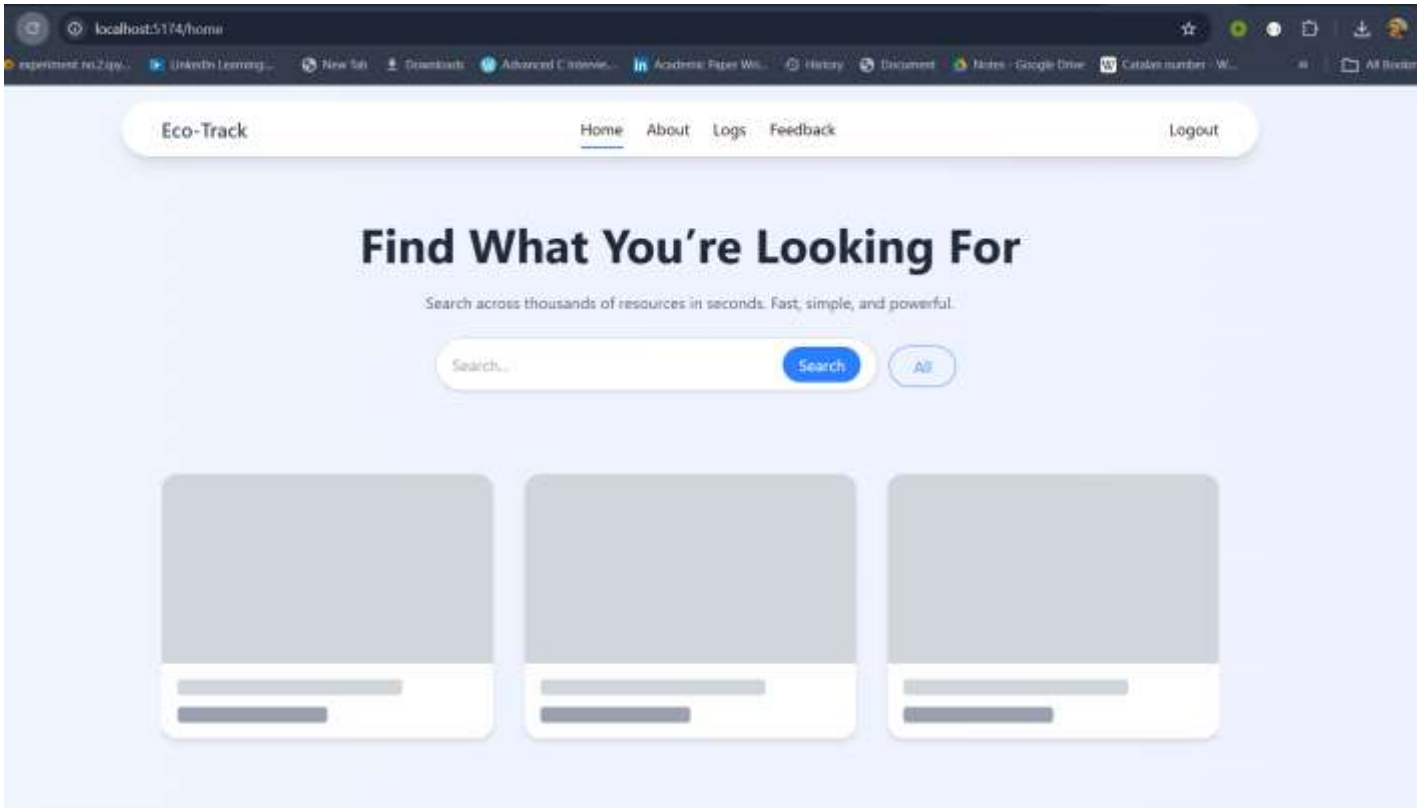
main.jsx

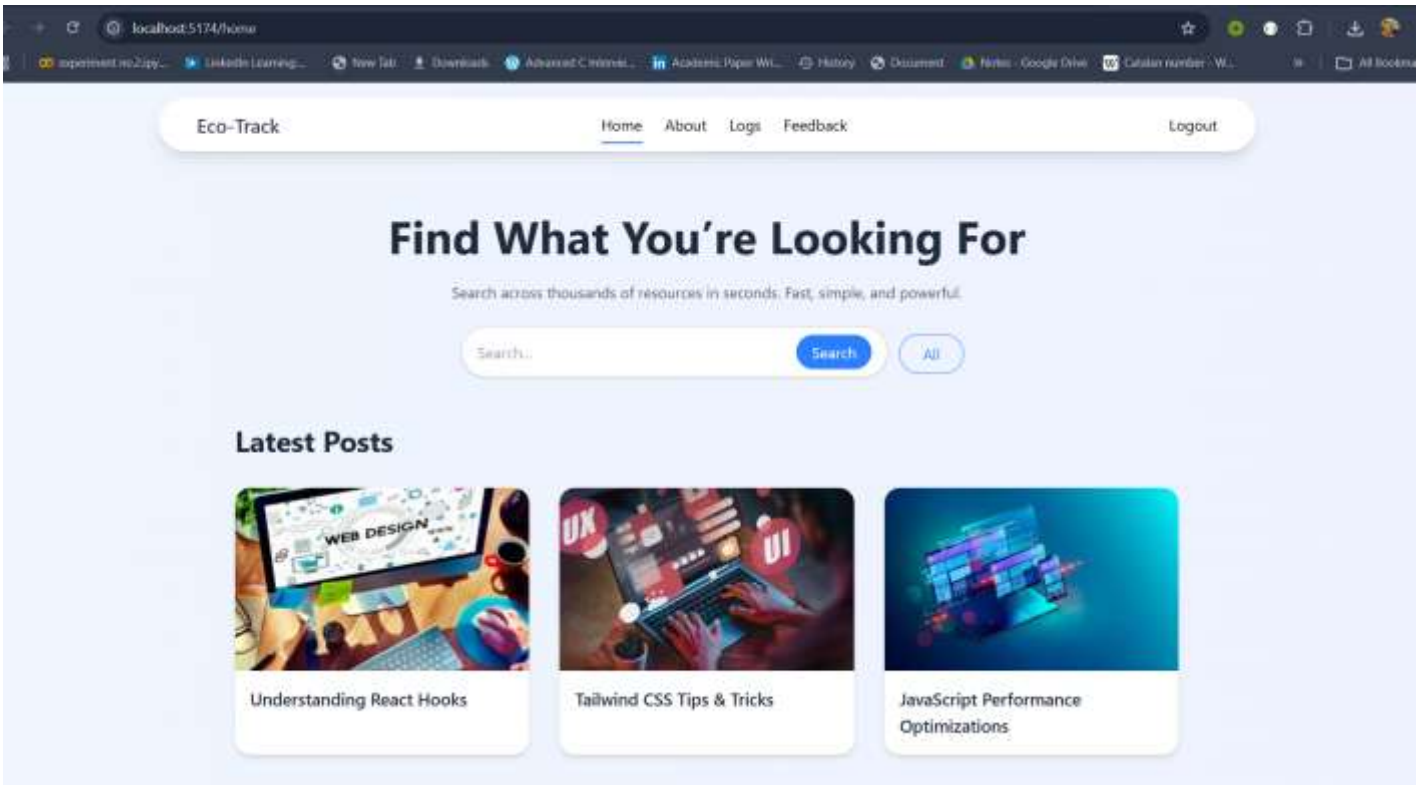
```jsx
import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import './index.css'
import App from './App.jsx'
// 1. BrowserRouter ko react-router-dom se import kar
import { BrowserRouter } from 'react-router-dom'
// 2. Provider aur Store ko import kar
import { Provider } from 'react-redux'
import store from './stores/store.js'
import { AuthProvider } from './context/AuthContext.jsx'

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

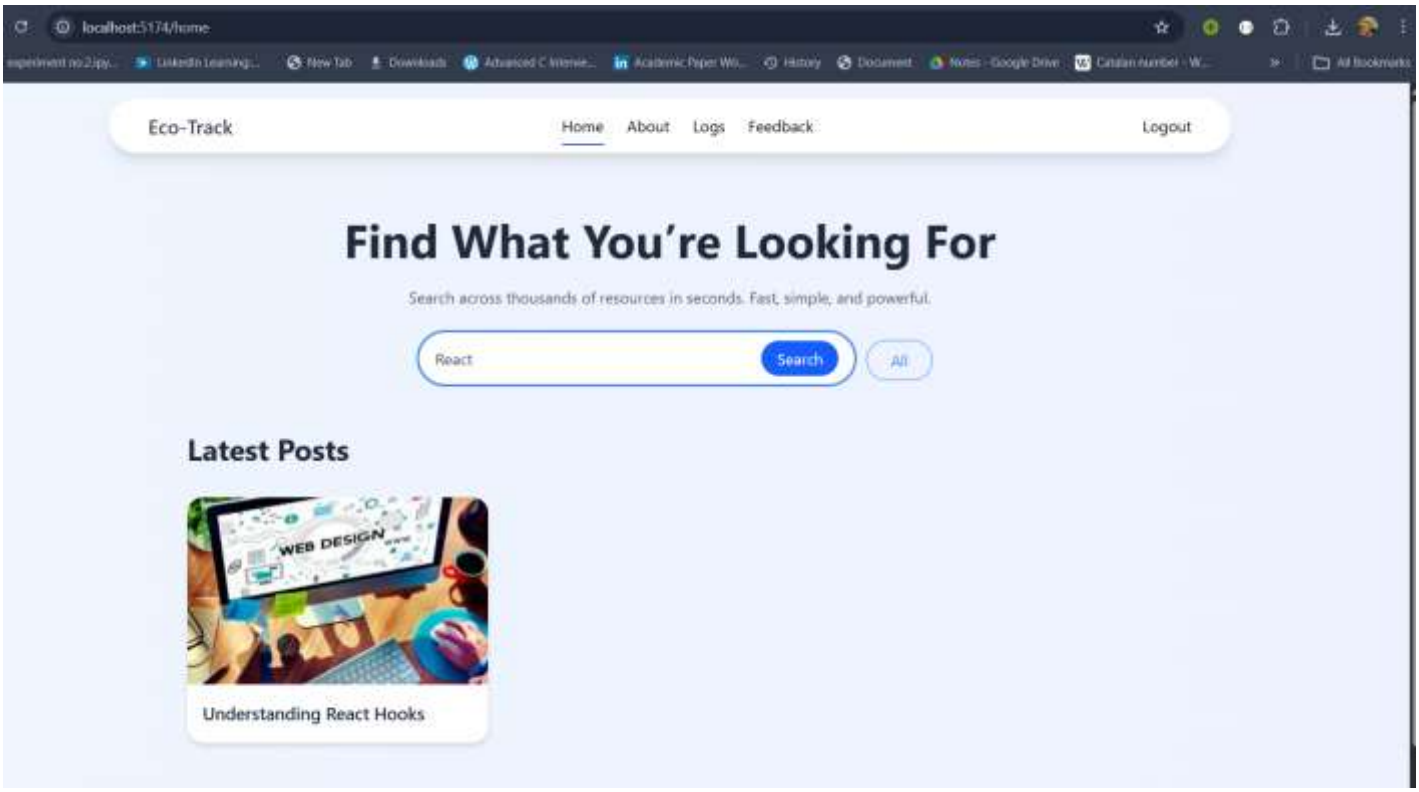
Logs.js

```jsx
import { useEffect } from "react";
import { useDispatch, useSelector } from "react-redux";
import { fetchLogs } from "../stores/logSlice";
import { calculateTotalCarbon } from '../data/Calculation.js';

function Logs() {
  const dispatch = useDispatch();

  // 1. Redux store se data, status aur error nikalna
  const { data, status, error } = useSelector((state) => state.logs);

  // 2. useEffect use karke data fetch karna (jaise image mein tha)
  useEffect(() => {
    if (status === "idle") {
      dispatch(fetchLogs());
    }
  }, [status, dispatch]);

  // 3. Loading aur Error handling
  if (status === "pending") {
    return <p style={{ textAlign: 'center', marginTop: '50px', fontSize: '40px', color: 'white' }}>Loading Logs ....
  }

  if (status === "failed") {
    return <p style={{ color: 'red', textAlign: 'center' }}>Error: {error}</p>;
  }

  // Calculation function ko ab Redux ke data ke saath call karenge
  const totalCarbon = data ? calculateTotalCarbon(data) : 0;

  return (
    <div style={{ color: 'black', background: 'white', height: '100%',width:'700px', border: '5px solid black', boxS
      <h1>Total Carbon Footprint: {totalCarbon}</h1>

      <h2>Activity Logs</h2>
      <table style={{ flex: 1, border: "2px solid black", margin: "auto", textAlign: "center", width: "100%" }}>
        <thead>
```

Output:







Learning Outcome:

Learnt about React Redux

Learnt about store , slice, reducer.

Learnt the usage of thunk Middleware.