



Experiment 4

Student Name: Ashutosh Yadav

Branch: CSE

Semester: 6th

Subject Name: Full Stack Development – II

UID: 23BCS11023

Section/Group: KRG-3-A

Date of Performance: 27/01/2026

Subject Code: 23CSH-309

1. Aim:

To optimize the performance of the EcoTrack React application using memoization techniques and code splitting, and to enhance the user interface using enterprise-grade Material UI components.

2. Objective:

1. Understand the causes of unnecessary re-renders in React applications
2. Optimize React components using React.memo to prevent avoidable re-renders
3. Apply useMemo to efficiently compute derived data and avoid redundant calculations
4. Use useCallback to memoize event handler functions and improve component performance
5. Implement lazy loading of components and routes using React.lazy and Suspense
6. Reduce initial bundle size and improve application load performance through code splitting
7. Enhance the visual appearance and usability of the EcoTrack application using Material UI components
8. Design a clean, consistent, and responsive user interface using Material UI layouts and typography

3. Implementation / Code:

Implementation Description:

In this experiment, the EcoTrack React application is optimized by identifying and eliminating unnecessary re-renders using memoization techniques such as React.memo, useMemo, and useCallback.

Computationally expensive operations and frequently passed callback functions are memoized to improve rendering efficiency. Code splitting is implemented using React.lazy and Suspense to enable lazy loading of components and routes, reducing the initial bundle size and improving load time. Additionally, the user interface is enhanced by integrating Material UI (MUI) components, leveraging its responsive layouts, typography, and theming system to create a clean, consistent, and enterprise-grade user experience.

Sample Code Snippet:



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

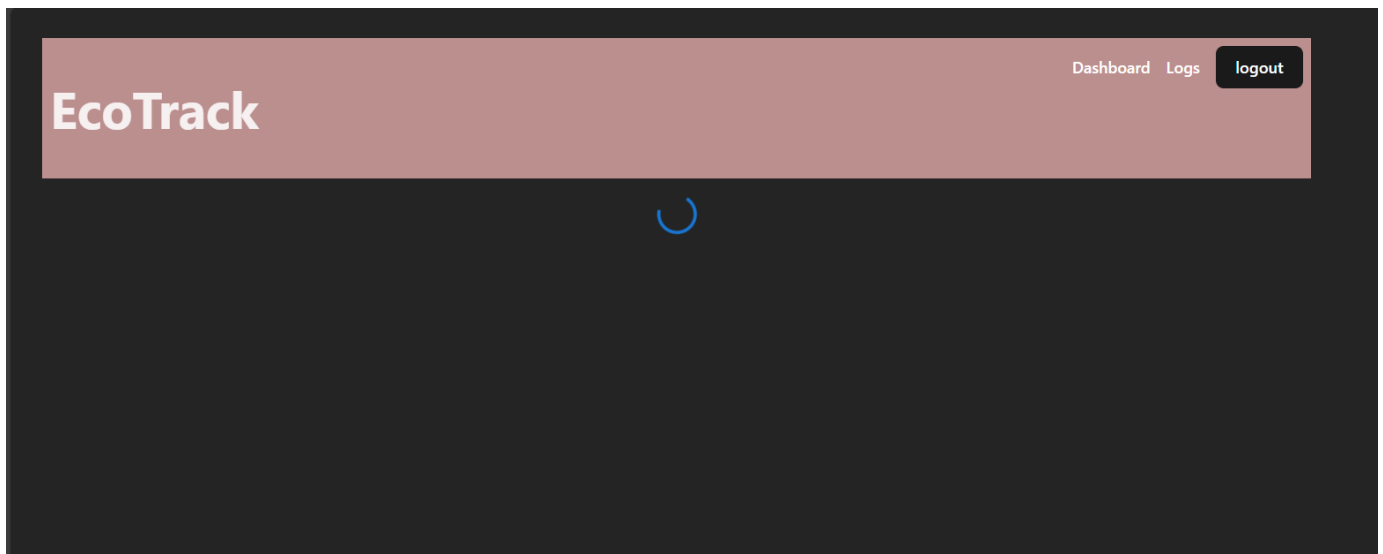
```
c > js App.js > App
1  import React, { Suspense } from "react";
2  import { BrowserRouter as Router, Routes, Route } from "react-router-dom";
3  import { CircularProgress, Box } from "@mui/material";
4
5  const Dashboard = React.lazy(() => import("../pages/Dashboard"));
6  const Dashboard2 = React.lazy(() => import("../pages/Dashboard2"));
7  const DashboardAnalytics = React.lazy(() =>
8  |   import("../pages/DashboardAnalytics")
9  | );
10 const DashboardLayout = React.lazy(() => import("../pages/DashboardLayout"));
11 const DashboardSummary = React.lazy(() => import("../pages/DashboardSummary"));
12 const Login = React.lazy(() => import("../pages/Login"));
13 const Logs = React.lazy(() => import("../pages/Logs"));
14
15 function App() {
16   return (
17     <Router>
18       <Suspense
19         fallback={
20           <Box
21             sx={{
22               display: "flex",
23               justifyContent: "center",+
24               justifyContent: "center",
25               alignItems: "center",
26               height: "100vh",
27             }}
28           <CircularProgress />
29         </Box>
30       </Suspense>
31     </Router>
32     <Routes>
33       <Route path="/login" element={ <Login /> } />
34       <Route path="/dashboard" element={ <Dashboard /> } />
35       <Route path="/summary" element={ <DashboardSummary /> } />
36       <Route path="/analytics" element={ <DashboardAnalytics /> } />
37     </Route>
38     <Route path="/dashboard2" element={ <Dashboard2 /> } />
39     <Route path="/dashboard-layout" element={ <DashboardLayout /> } />
40     <Route path="/summary" element={ <DashboardSummary /> } />
41     <Route path="/analytics" element={ <DashboardAnalytics /> } />
42   </Route>
43   <Route path="/logs" element={ <Logs /> } />
44 </Routes>
45 </Suspense>
46 </Router>
47 );
```

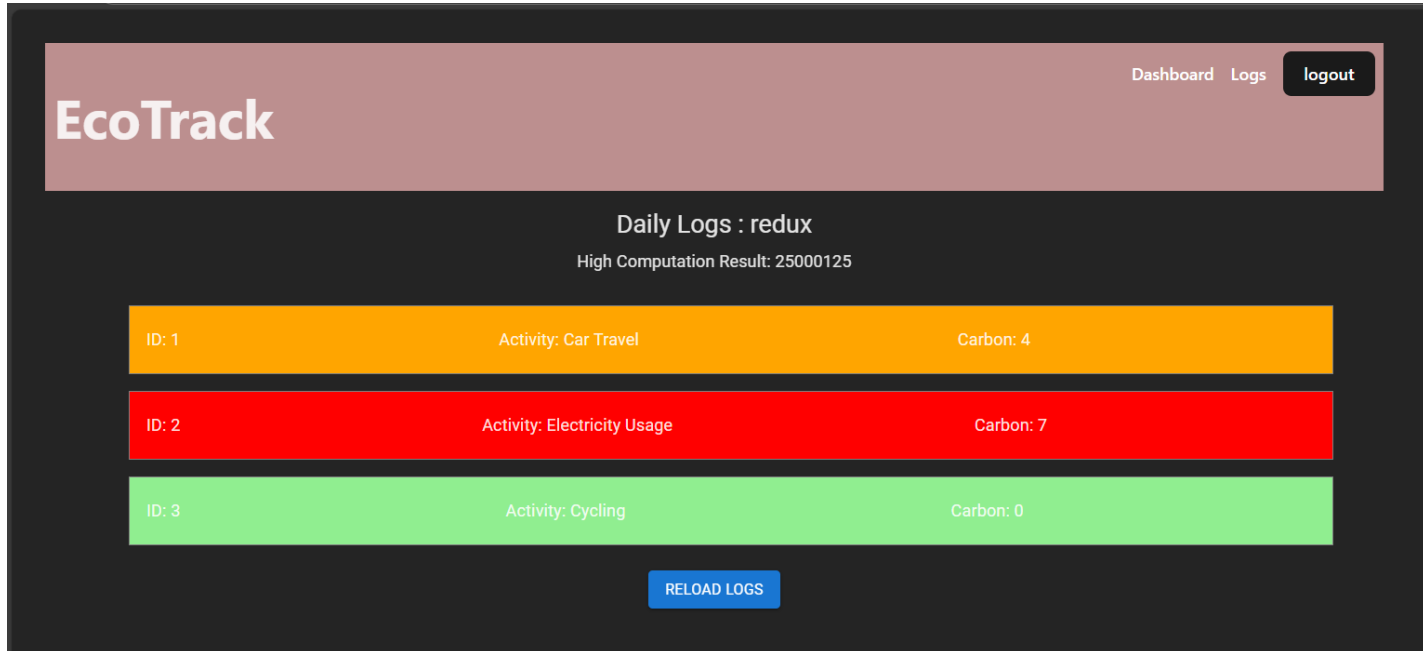
```
src > pages > Logs.jsx > ...
1  import { useEffect, useCallback, useMemo } from "react";
2  import { useDispatch, useSelector } from "react-redux";
3  import { fetchLogs } from "../store/logSlice";
4  import Log from "../Log";
5  import {
6    Container,
7    Typography,
8    List,
9    Button,
10   CircularProgress,
11   Box,
12 } from "@mui/material";
13
14  const Logs = () => {
15    const dispatch = useDispatch();
16    const { data, status } = useSelector((state) => state.logs);
17
18    useEffect(() => {
19      if (status === "idle") {
20        dispatch(fetchLogs());
21      }
22    }, [status, dispatch]);
23
24    const expensiveValue = useMemo(() => {
25      let result = 0;
26      for (let i = 0; i < 50000000; i++) {
27        result += Math.random();
28      }
29      return result.toFixed(0);
30    }, []);
31
32    const handleRefresh = useCallback(() => {
33      dispatch(fetchLogs());
34    }, [dispatch]);
35
36    if (status === "loading") {
37      return (
38        <Box sx={{ display: "flex", justifyContent: "center", padding: "1rem" }}>
39          <CircularProgress />
40        </Box>
41      );
42    }
43
44    return (
45      <Container sx={{ padding: "1rem" }}>
46        <Typography variant="h5" gutterBottom>
47          Daily Logs : redux
48        </Typography>
49        <Typography variant="body1" gutterBottom>
50          High Computation Result: {expensiveValue}
51        </Typography>
52        <List>
53          {data.map((log) => (
54            <Log key={log.id} id={log.id} activity={log.activity} carbon={log.carbon} />
55          ))}
56        </List>
57        <Button variant="contained" onClick={handleRefresh}>
58          Reload logs
59        </Button>
60      </Container>
61    );
62  };
63
64  export default Logs;
```

```
src > pages > Log.jsx > Log
1  import React from "react";
2  import { ListItem, ListItemText, Box } from "@mui/material";
3
4  const Log = ({ id, activity, carbon }) => {
5    const backgroundColor =
6      carbon <= 3 ? "lightgreen" : carbon <= 6 ? "orange" : "red";
7
8    return (
9      <ListItem
10        sx={{
11          backgroundColor,
12          border: "1px solid grey",
13          padding: "1rem",
14          margin: "1rem",
15          display: "flex",
16          gap: "2rem",
17        }}
18      >
19        <ListItemText primary={`ID: ${id}`} />
20        <ListItemText primary={`Activity: ${activity}`} />
21        <ListItemText primary={`Carbon: ${carbon}`} />
22      </ListItem>
23    );
24  };
25
26  export default React.memo(Log);
```

4. Output:

- Improved the runtime performance of the EcoTrack React application by minimizing unnecessary re-renders.
- Achieved faster initial load time and better scalability through effective code splitting and lazy loading.
- Developed a maintainable and performance-optimized React codebase using memoization best practices.
- Enhanced user experience with a clean, responsive, and enterprise-grade UI built using Material UI components.





The screenshot shows the EcoTrack application dashboard. At the top, there is a navigation bar with links for 'Dashboard', 'Logs', and a 'logout' button. The main heading is 'EcoTrack'. Below this, the section is titled 'Daily Logs : redux' with a subtitle 'High Computation Result: 25000125'. The logs are displayed as a list of three items, each in a colored box: ID: 1 (orange) for 'Car Travel' with a carbon value of 4; ID: 2 (red) for 'Electricity Usage' with a carbon value of 7; and ID: 3 (green) for 'Cycling' with a carbon value of 0. A 'RELOAD LOGS' button is located at the bottom center of the dashboard.

ID	Activity	Carbon
ID: 1	Activity: Car Travel	Carbon: 4
ID: 2	Activity: Electricity Usage	Carbon: 7
ID: 3	Activity: Cycling	Carbon: 0

5. Learning Outcomes

- Understand how unnecessary re-renders impact React application performance and how to prevent them.
- Apply memoization techniques (`React.memo`, `useMemo`, `useCallback`) to optimize component rendering and computations.
- Implement code splitting and lazy loading to reduce bundle size and improve application load performance.
- Design a responsive, professional UI using Material UI components, layouts, and typography.