

Experiment 4

Student Name: Analava Bera

Branch: BE CSE

Semester: 6th

Subject Name: Full Stack Development

UID: 23BCS13611

Section/Group: KRG 3A

Date of Performance: 03/02/26

Subject Code: 23CSH-309

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.

Objective:

After completing this experiment, the student will be able to:

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

Implementation/Code:

Login.jsx:

```
import { useAuth } from "../context/AuthContext";
import { useNavigate } from "react-router-dom";
import Button from '@mui/material/Button';
import Box from '@mui/material/Box';
import Typography from '@mui/material/Typography';
```

```
const Login = () => {
```

```
const { login } = useAuth(); const
navigate = useNavigate(); const
handleLogin = () => { login();
  navigate("/dashboard");
};

return (
  <Box sx={{
    display: 'flex',
    flexDirection: 'column',
    alignItems: 'center',
    justifyContent: 'center',
    minHeight: '100vh',
    p: 3
  }}>
    <Typography variant="h4" gutterBottom>
      Welcome to EcoTrack
    </Typography>
    <Button
      variant="contained"
      color="primary"
      size="large"
      onClick={handleLogin}
      sx={{ mt: 3, px: 6, py: 1.5 }}
    >
      Login to EcoTrack
    </Button>
  </Box>
);
};
```

export default Login;

Dashboard.jsx:

```
import { useState } from 'react';
import { Outlet, useNavigate, Link as RouterLink } from 'react-router-dom'; import
{ useAuth } from "../context/AuthContext";
import Header from "../components/Header";
```

```
import AppBar from '@mui/material/AppBar'; import
Box from '@mui/material/Box'; import Toolbar from
 '@mui/material/Toolbar'; import Typography from
 '@mui/material/Typography'; import Drawer from
 '@mui/material/Drawer'; import List from
 '@mui/material/List'; import ListItem from
 '@mui/material/ListItem'; import ListItemButton from
 '@mui/material/ListItemButton'; import ListItemIcon
 from '@mui/material/ListItemIcon'; import
 ListItemText from '@mui/material/ListItemText';
import Divider from '@mui/material/Divider'; import
 IconButton from '@mui/material/IconButton';
import Button from '@mui/material/Button';

import HomeIcon from '@mui/icons-material/Home'; import
 DashboardIcon from '@mui/icons-material/Dashboard'; import
 AssessmentIcon from '@mui/icons-material/Assessment'; import
 LogoutIcon from '@mui/icons-material/Logout';
import MenuIcon from '@mui/icons-material/Menu';

const drawerWidth = 240;

const Dashboard = () => {
  const { logout } = useAuth();
  const navigate = useNavigate();
  const [mobileOpen, setMobileOpen] = useState(false);

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

  const handleDrawerToggle = () => {
    setMobileOpen(!mobileOpen);
  };

  const drawerContent = (
    <div>
      <Toolbar /> { /* spacer under AppBar */}
      <Divider />
```

```

<List>
  <ListItem disablePadding>
    <ListItemButton
      component={RouterLink}
      to=""
      onClick={() => setMobileOpen(false)}
    >
      <ListItemIcon><HomeIcon /></ListItemIcon>
      <ListItemText primary="Home" />
    </ListItemButton>
  </ListItem>

  <ListItem disablePadding>
    <ListItemButton
      component={RouterLink}
      to="overview"
      onClick={() => setMobileOpen(false)}
    >
      <ListItemIcon><DashboardIcon /></ListItemIcon>
      <ListItemText primary="Overview" />
    </ListItemButton>
  </ListItem>

  <ListItem disablePadding>
    <ListItemButton
      component={RouterLink}
      to="reports"
      onClick={() => setMobileOpen(false)}
    >
      <ListItemIcon><AssessmentIcon /></ListItemIcon>
      <ListItemText primary="Reports" />
    </ListItemButton>
  </ListItem>
</List>
<Divider />
<List>
  <ListItem disablePadding>
    <ListItemButton onClick={handleLogout} sx={{ color: 'error.main' }}>
      <ListItemIcon><LogoutIcon color="error" /></ListItemIcon>
      <ListItemText primary="Logout" />
    </ListItemButton>
  </ListItem>
</List>

```

```

        </ListItemButton>
      </ListItem>
    </List>
  </div>
);

return (
  <Box sx={{ display: 'flex' }}>
    { /* AppBar at top */ }
    <AppBar
      position="fixed" sx={{
        width: { sm: `calc(100% - ${drawerWidth}px)` },
        ml: { sm: `${drawerWidth}px` }, // ← important: push content right
        backgroundColor: '#4CAF50',
      }}
    >
      <Toolbar>
        <IconButton
          color="inherit"
          edge="start"
          onClick={handleDrawerToggle}
          sx={{ mr: 2, display: { sm: 'none' } }}
        >
          <MenuIcon />
        </IconButton>
        <Typography variant="h6" noWrap component="div">
          Dashboard
        </Typography>
      </Toolbar>
    </AppBar>

    { /* Sidebar - LEFT side */ }
    <Box component="nav"
      sx={{
        width: { sm:
          drawerWidth },
        flexShrink:
          { sm: 0 }
      }}
    >
      { /* Mobile temporary drawer */ }

```

```

<Drawer
variant="temporary"
open={mobileOpen}
  onClose={handleDrawerToggle}
  ModalProps={{ keepMounted: true }} sx={{
    display: { xs: 'block', sm: 'none' },
    '& .MuiDrawer-paper': {
      boxSizing: 'border-box',      width:
drawerWidth,
      backgroundColor: '#4CAF50', // green theme
      color: 'white',
    },
  }}
>
  {drawerContent}
</Drawer>

```

```

{/* Desktop permanent drawer - LEFT */}
<Drawer
  variant="permanent"
  sx={{
    display: { xs: 'none', sm: 'block' },
    '& .MuiDrawer-paper': {
      boxSizing: 'border-box',      width:
drawerWidth,
      backgroundColor: '#4CAF50',
      color: 'white',
    },
  }}
  open
  >
    {drawerContent}
  </Drawer>
</Box>

```

```

{/* Main content - pushed to right */}
<Box
  component="main"
  sx={{

```

```

    flexGrow: 1,
  p: 3,
    width: { sm: `calc(100% - ${drawerWidth}px)` },      ml: {
sm: `${drawerWidth}px` },    // ← ensures shift      mt: '64px',
// space under AppBar (Toolbar height)      backgroundColor:
'background.default',
    }}
  >
    { /* Spacer for fixed AppBar */ }
  <Toolbar />

  { /* This is where your child routes render: DashboardHome / Overview / Reports */ }
  <Outlet />
</Box>
</Box>
);
};

```

export default Dashboard;

App.jsx:

```

import { lazy, Suspense } from "react";      import {
Routes, Route, Navigate } from "react-router-dom"; import
ProtectedRoute from "./routes/ProtectedRoute";

// Lazy load pages
const Dashboard    = lazy(() => import("./pages/Dashboard")); const
DashboardHome    = lazy(() => import("./pages/DashboardHome")); const
Overview        = lazy(() => import("./pages/Overview")); const Reports
= lazy(() => import("./pages/Reports")); const Login          = lazy(() =>
import("./pages/Login"));

const App = () => {
  return (
    <Suspense fallback={
  <div style={{      display:
'flex',      justifyContent:
'center',      alignItems:
'center',      minHeight:
'100vh'

```

```

    }}>
    Loading application...
  </div>
}>
<Routes>
  <Route path="/login" element={<Login />} />

  <Route
    path="/dashboard"
    element={
      <ProtectedRoute>
        <Dashboard />
      </ProtectedRoute>
    }
  >
    <Route index element={<DashboardHome />} />
    <Route path="overview" element={<Overview />} />
    <Route path="reports" element={<Reports />} />
  </Route>

  <Route path="*" element={<Navigate to="/dashboard" />} />
</Routes>
</Suspense>
);
};

export default App;

```

Main.jsx: import React from "react"; import
ReactDOM from "react-dom/client"; import App from
"./App"; import { BrowserRouter } from "react-router-
dom"; import { AuthProvider } from
"./context/AuthContext"; import { Provider } from
"react-redux"; import { store } from "./redux/store";
import { ThemeProvider } from
'@mui/material/styles'; import CssBaseline from
'@mui/material/CssBaseline'; import theme from
"./theme"; import "./index.css"; import
"@fontsource/roboto/300.css"; import
"@fontsource/roboto/400.css"; import


```
"@fontsource/roboto/500.css"; import  
"@fontsource/roboto/700.css";
```

```
ReactDOM.createRoot(document.getElementById("root")).render(  
  <React.StrictMode>  
    <Provider store={store}>  
      <ThemeProvider theme={theme}>  
        <CssBaseline />  
        <BrowserRouter>  
          <AuthProvider>  
            <App />  
          </AuthProvider>  
        </BrowserRouter>  
      </ThemeProvider>  
    </Provider>  
  </React.StrictMode>  
);
```

Output:

Welcome to EcoTrack

Login to EcoTrack

Dashboard



Home



Overview



Reports



Logout

Total Activities

- Car Travel: 4 Kg
- Electricity Usage: 6 Kg
- Cycling: 0 Kg

High Carbon (> 4 Kg)

- Electricity Usage

Low Carbon (≤ 4 Kg)

- Car Travel
- Cycling

Refresh Logs

Overview Page

This is the Overview section of EcoTrack.

Reports & Analytics

Detailed reports and carbon footprint analysis coming soon...

Learning Outcome:

1. Identify and analyze causes of unnecessary re-renders in a React application.
2. Apply React.memo, useMemo, and useCallback to optimize component rendering performance.
3. Implement code splitting and lazy loading using React.lazy and Suspense to improve load time.
4. Design a responsive and consistent UI using Material UI components and theming.
5. Integrate performance optimization techniques with secure routing and modern React architecture.