DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
Discover. Learn. Empower.

Experiment 5

Student Name: Harshdeep Singh
UID: 23BAI70623
Branch: BE CSE AIML
Section/Group: 23AIT-KRG G1
Semester: 6th
Date of Performance: 18 Feb 2026
Subject Name: Full Stack
Subject Code: 23CSH-382

1. **Title: Application Testing & Debugging**

2. **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.

3. **Objective:**
   After completing this experiment and its follow-up tasks, the student will be able to:
   1. Understand the purpose of automated testing in frontend applications
   2. Write unit tests for JavaScript utility functions using Jest
   3. Use different Jest matchers to validate expected outputs and behaviors
   4. Test React components using React Testing Library
   5. Verify UI rendering by querying elements from the DOM
   6. Implement asynchronous testing using findBy and waitFor methods
   7. Apply mocking to simulate API or external data responses in tests
   8. Perform snapshot testing to detect unintended UI changes
   9. Debug failing tests and application logic using browser Developer Tools and breakpoints
   10. Analyze application behavior and errors systematically rather than manual checking.

4. Implementation/Code:

- logSlice.jsx:

```jsx
import { createSlice, createAsyncThunk } from "@reduxjs/toolkit";

export const initialState = {
  data: [],
  status: "idle",
  error: null,
};

export const fetchLogs = createAsyncThunk(
  "logs/fetchLogs",
  async (_, { rejectWithValue }) => {
    try {
      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 },
      ];
    } catch (error) {
      return rejectWithValue("Failed to fetch logs");
    }
  }
);

const logsSlice = createSlice({
  name: "logs",
  initialState,
  reducers: {
    resetLogs: () => initialState,
  },
  extraReducers: (builder) => {
    builder
      .addCase(fetchLogs.pending, (state) => {
        state.status = "loading";
        state.error = null;
      })
      .addCase(fetchLogs.fulfilled, (state, action) => {
        state.status = "success";
        state.data = action.payload;
      })
      .addCase(fetchLogs.rejected, (state, action) => {
        state.status = "failed";
        state.error = action.payload || action.error.message;
      });
  },
});

export const { resetLogs } = logsSlice.actions;
export default logsSlice.reducer;
```

- Logs.jsx:

```jsx
import { useEffect, useCallback } from "react";
import { useDispatch, useSelector } from "react-redux";
import { fetchLogs } from "../store/logSlice";
import LoadingSpinner from "../components/Loading";
import CarbonCard from "../components/CarbonCard";
import { calculateTotalCarbon } from "../utils/carbon";

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

  const handleRefresh = useCallback(() => {
    dispatch(fetchLogs());
  }, [dispatch]);

  useEffect(() => {
    if (status === "idle") {
      dispatch(fetchLogs());
    }
  }, [status, dispatch]);

  if(status === "loading"){
    return <LoadingSpinner />;
  }
  if(status === "failed"){
    return <p>Error: {error}</p>;
  }
  const totalCarbon = calculateTotalCarbon(data);

  return (
    <div>
      <h1>Logs</h1>

      {data.map((log) => (
        <CarbonCard
          key={log.id}
          activity={log.activity}
          carbon={log.carbon}
        />
      ))}

      <h3 data-testid="total-carbon">
        Total Carbon: {totalCarbon} kg CO2
      </h3>

      <button onClick={handleRefresh}>Refresh</button>
    </div>
  );
};

export default Logs;
```

- CarbonCards.jsx:

```jsx
const CarbonCard = ({ activity, carbon }) => {
  return (
    <div>
      <h4>{activity}</h4>
      <p>{carbon} kg CO2</p>
    </div>
  );
};

export default CarbonCard;
```

- Loading.jsx:

```jsx
const Loading = () => {
  return <p>Loading...</p>;
};

export default Loading;
```

- NotFound.jsx:

```jsx
const NotFound = () => {
  return <h2>404 - Page Not Found</h2>;
};

export default NotFound;
```

- Store.jsx:

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

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

export default store;
```

- AuthContext.jsx:

```jsx
import { createContext, useContext, useState } from "react";

const AuthContext = createContext(null);

export const AuthProvider = ({children}) => {
    const [isAuthenticated, setIsAuthenticated] = useState(false);

    return (
        <AuthContext.Provider value = {{isAuthenticated, setIsAuthenticated}}>
            {children}
        </AuthContext.Provider>
    )
}

export const useAuth = () => useContext(AuthContext);
```

- ProtectedRoute.jsx:

```jsx
import { Navigate } from "react-router-dom";
import { useAuth } from "../context/AuthContext";
import { children } from "react";

const ProtectedRoute = ({children}) => {
    const {isAuthenticated} = useAuth();

    if(!isAuthenticated) {
        return <Navigate to = "/login" replace/>
    }
    return children;
}

export default ProtectedRoute;
```

- Login.jsx:

```jsx
import { useAuth } from "../context/AuthContext";
import { useNavigate } from "react-router-dom";

const Login = () => {
    const {setIsAuthenticated} = useAuth();
    const navigate = useNavigate();

    const handleLogin = () => {
        setIsAuthenticated(true);
        navigate("/");
    }
    return (
        <>
        <h3>Login</h3>
        <button data-testid="login-button" onClick={handleLogin}>Login</button>
        </>
    )
}
export default Login;
```

- Logout.jsx:

```jsx
import { useEffect } from "react";
import { useNavigate } from "react-router-dom";
import { useAuth } from "../context/AuthContext";

const Logout = () => {
  const { setIsAuthenticated } = useAuth();
  const navigate = useNavigate();

  useEffect(() => {
    setIsAuthenticated(false);
    navigate("/login");
  }, [setIsAuthenticated, navigate]);

  return <p>Logging you out...</p>;
}

export default Logout;
```

- DashboardSettings, DashboardSummary, DashboardAnalytics:

```jsx
const DashboardSettings = () => {
    return (
        <h3>These are the settings</h3>
    )
}

export default DashboardSettings;
```

```jsx
const DashboardSummary = () => {
    return (
        <h3>This is a Summary</h3>
    )
}

export default DashboardSummary;
```

```jsx
import { useSelector } from "react-redux";
import { calculateTotalCarbon, classifyCarbon } from "../utils/carbon";

const DashboardAnalytics = () => {
  const logs = useSelector((state) => state.logs.data);

  const total = calculateTotalCarbon(logs);
  const category = classifyCarbon(total);

  return (
    <div>
      <h3>Analytics</h3>
      <p>Total Carbon: {total} kg CO2</p>
      <p>Impact Level: {category}</p>
    </div>
  );
};

export default DashboardAnalytics;
```

- DashboardLayout.jsx:

```jsx
import { Link, Outlet } from "react-router-dom";
import { Container, Button, Stack, Typography } from "@mui/material";

const DashboardLayout = () => {
  return (
    <Container>
      <Typography variant="h4" gutterBottom>
        Dashboard
      </Typography>

      <Stack direction="row" spacing={2} marginBottom={2}>
        <Button variant="contained" component={Link} to="settings">
          Settings
        </Button>

        <Button variant="contained" component={Link} to="summary">
          Summary
        </Button>

        <Button variant="contained" component={Link} to="analytics">
          Analytics
        </Button>
      </Stack>

      <Outlet />
    </Container>
  );
};

export default DashboardLayout;
```

- Header.jsx:

```jsx
import React from "react";
import { AppBar, Toolbar, Typography, Button } from "@mui/material";
import { Link } from "react-router-dom";
import { useAuth } from "../context/AuthContext";

const Header = () => {
  const { isAuthenticated } = useAuth();

  return (
    <AppBar position="static">
      <Toolbar>
        <Typography variant="h6" sx={{ flexGrow: 1 }}>
          EcoTrack
        </Typography>

        <Button color="inherit" component={Link} to="/">
          Dashboard
        </Button>

        <Button color="inherit" component={Link} to="/logs">
          Logs
        </Button>

        {isAuthenticated ? (
          <Button color="inherit" component={Link} to="/logout">
            Logout
          </Button>
        ) : (
          <Button color="inherit" component={Link} to="/login">
            Login
          </Button>
        )}
      </Toolbar>
    </AppBar>
  );
};

export default React.memo(Header);
```

- App.jsx:

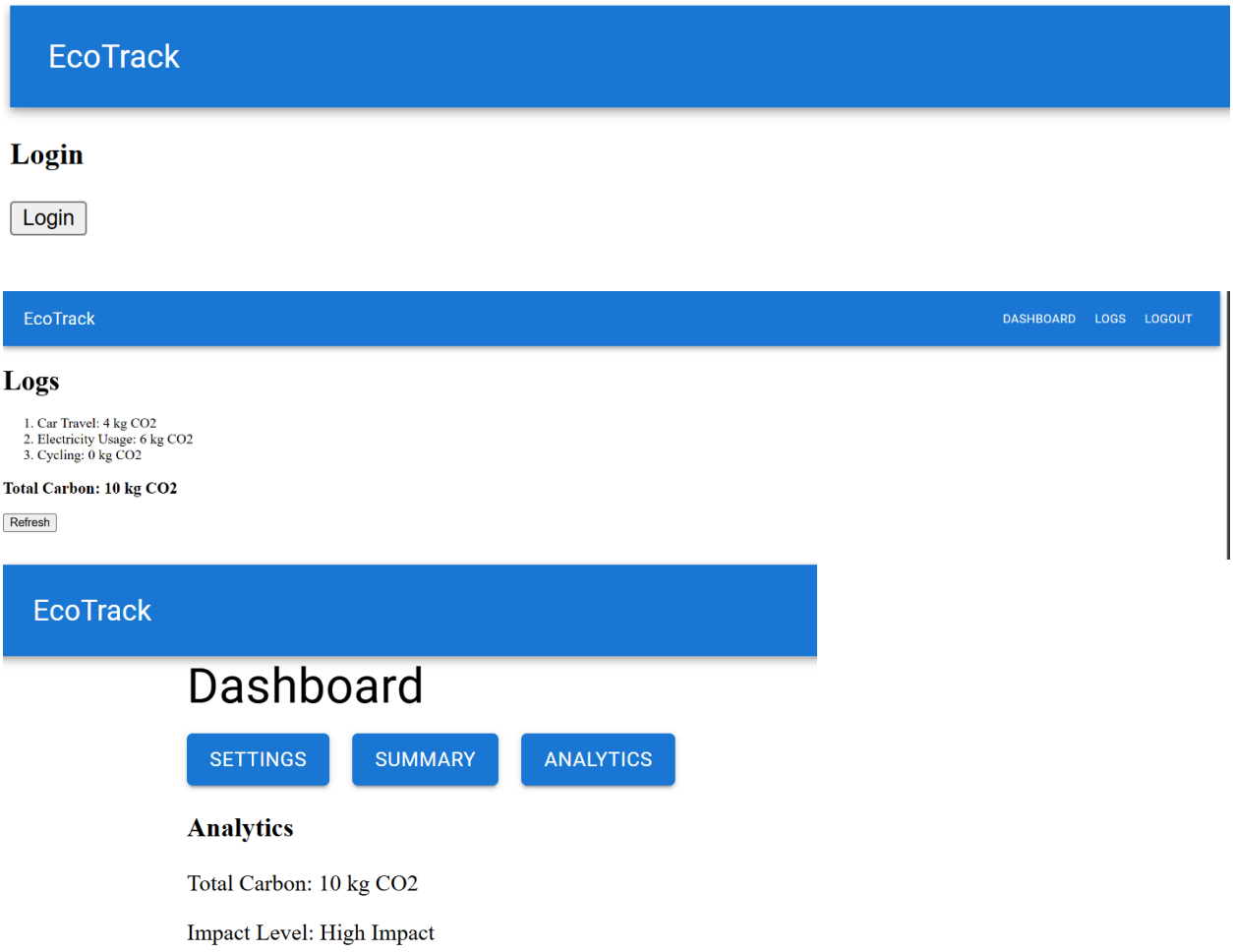
```jsx
import { Route, Routes } from "react-router-dom";
import { Suspense, lazy } from "react";
import ProtectedRoute from "./routes/ProtectedRoute";
import Header from "./components/Header";
import NotFound from "./pages/NotFound";

const Login = lazy(() => import("./pages/Login"));
const Logout = lazy(() => import("./pages/Logout"));
const DashboardLayout = lazy(() => import("./pages/DashboardLayout"));
const DashboardSummary = lazy(() => import("./pages/DashboardSummary"));
const DashboardSettings = lazy(() => import("./pages/DashboardSettings"));
const DashboardAnalytics = lazy(() => import("./pages/DashboardAnalytics"));
const Logs = lazy(() => import("./pages/Logs"));

function App() {
  return (
    <>
      <Header />
      <Suspense fallback={<h2>Loading...</h2>}>
        <Routes>
          <Route path="/login" element={<Login />} />
          <Route path="/logout" element={<Logout />} />
          <Route
            path="/"
            element={
              <ProtectedRoute>
                <DashboardLayout />
              </ProtectedRoute>
            }
          >
            <Route index element={<DashboardSummary />} />
            <Route path="settings" element={<DashboardSettings />} />
            <Route path="summary" element={<DashboardSummary />} />
            <Route path="analytics" element={<DashboardAnalytics />} />
            <Route path="*" element={<NotFound />} />
          </Route>
          <Route
            path="/logs"
            element={
              <ProtectedRoute>
                <Logs />
              </ProtectedRoute>
            }
          />
        </Routes>
      </Suspense>
    </>
  );
}

export default App;
```

5. Output



6. Learning Outcome

- We learnt about React Apps and how to create them.
- We learnt about redux and it's components.
- We learnt about the use of thunks and Slices.
- We learnt about Authentication and index pages
- We learnt about React testing library and its components.
- We learnt about the use and implementation of snapshot testing.