**Advanced React**

Course-End Project Problem Statement



**Create Expense Management Application**

## **Project objective:**

To develop an expense management application for optimizing telecom usage optimization using React and Redux with advanced concepts. The application will analyze customer telecom usage, suggest optimized plans based on historical data, provide real-time cost-saving tips, and allow users to make instant plan adjustments.

The goal is to create an efficient and user-friendly application using cutting-edge web technologies to enhance the telecom user experience.

## **Background of the problem statement:**

NetCarrier, a telecom company, aims to provide a solution to customers who often encounter challenges in managing and optimizing their usage plans.

Without real-time insights and personalized recommendations, users may not be aware of more cost-effective plans or may miss opportunities to adjust their plans based on changing usage patterns.

This project addresses these challenges by providing an application that analyzes historical usage data, offers real-time suggestions, and allows users to dynamically adjust their plans for optimal cost savings.

## **Implementation requirements:**

The expected deliverables are as follows:

1. Develop a user-friendly UI using React and Create React App. Design a dashboard that intuitively presents usage data and recommendations.
2. Implement navigation using React Router, enabling users to switch between different views seamlessly.
3. Utilize Redux with the Redux toolkit for efficient state management, ensuring a centralized and consistent data flow.
4. Implement real-time updates to provide users with the latest information on their telecom usage.
5. Integrate with simulated backend APIs to fetch historical usage data and receive real-time updates.

## **You must use the following:**

1. React with Create React App for building the user interface.
2. Redux with Redux toolkit for efficient state management.
3. React Router for handling navigation between different views.
4. Redux Thunk for handling asynchronous actions, especially for API calls.
5. Axios or Fetch for making API calls to simulate fetching customer data.
6. useAsync and useRealTimeUpdates custom Hooks for managing asynchronous data and real-time updates.
7. withDataFetching for wrapping components that require data fetching, such as Higher Order Components.
8. Advanced React features such as useMemo to optimize rendering where applicable.
9. Redux DevTools integration for debugging and inspecting actions and state changes.
10. UsageChart component implementation with performance optimization using useMemo.

## **The following requirements should be met:**

1. Redux store:  
   Set up a Redux store in your application using the configureStore method to define the store's behavior and initial state.
2. User authentication:

Allow users to log in and authenticate to access personalized usage data and recommendations.

1. Navigation:

Implement smooth navigation between different views, ensuring a seamless user experience.

1. Dashboard overview:

Display a comprehensive dashboard providing an overview of historical usage data and current plan details.

1. Usage analysis charts:

Create interactive charts and graphs for visualizing historical usage data, helping users understand their usage patterns.

1. Recommendations section:

Provide a dedicated section for personalized recommendations based on historical data analysis.

1. Plan adjustment interface:

Enable users to adjust their telecom plans directly from the dashboard, providing instant plan customization.

1. Performance optimization:

Optimize components for performance using techniques like useMemo and React memo.

**API Endpoints:**

|  |  |  |
| --- | --- | --- |
| Action | Method | Endpoint |
| Login API | POST | <http://localhost:9010/login> Test credentials: john@example.com | john123 fionna@example.com | fionna123 sia@example.com | sia123 |
| Retrieve plan for a user | GET | <http://localhost:9010/get-user-plan>  http://localhost:9010/get-user-plan?email=john@example.com |
| Retrieve user plan usage records of a user | GET | <http://localhost:9010/get-user-plan-usage>  http://localhost:9010/get-user-plan-usage?email=john@example.com |
| Update user plan | POST | <http://localhost:9010/update-user-plan> email: john@example.com  plan id: P102 |
| Retrieve all telecom plans | GET | <http://localhost:9010/get-plans> |

**Backend project to be used:**

Download the demoapi-v1.jar file from [link](https://github.com/GithubWorkstation/Adv_React_JAR).

Execute the below command to bring up the backend:

**java -jar demoapi-v1.jar**