Assignnment-11

Name: Thumiki Satya Sree
Hall Ticket No:2303A51222
Batch No:04

1. Aim of the Assignment

The aim of this assignment is to connect a React frontend with a Node.js/Express backend API to fetch and display financial summary information such as total income, total expenses, and balance on the Dashboard page.

2. Technologies Used

1. Frontend: React.js
2. Backend: Node.js, Express.js
3. API Communication: Axios
4. Tools: VS Code, Node Package Manager (npm)

3.Frontend Dashboard Functionality

1) The Dashboard page in React is designed to:
2) Automatically request data from the backend when the page loads.
3) Store the received data in component state.

Display:

I. Total Income
II. Total Expenses
III. Balance (Income − Expenses)

The dashboard updates dynamically based on the data received from the API.

4.Data Fetching Process

When the Dashboard component loads:

1. A request is sent to the backend API.
2. The backend responds with financial summary data.
3. The frontend stores this data in state variables.
4. The UI updates automatically to display the values.

This process happens without refreshing the browser page.

5.Bonus Features

5.1 Refresh Dashboard Without Reloading

A refresh option is provided to:

1) Fetch updated dashboard data again.
2) Update the displayed values instantly.
3) Avoid full page reloads.

This makes the application more interactive and efficient.

5.2 Currency Formatting

Financial values are displayed in a proper currency format:

4) Numbers are formatted with commas.
5) Currency symbol is shown for better readability.
6) This improves clarity and professional appearance of the dashboard.

Expected Outcome

1. Dashboard data is loaded dynamically from the backend.
2. No page reload occurs during data fetching.
3. Total income, total expenses, and balance are displayed correctly.
4. User receives proper feedback during loading and error situations.

Dashboard

Total Income: 5000

Total Expenses: 3000

Balance: 2000

Conclusion-

This assignment demonstrates how a React frontend can effectively communicate with a Node.js backend using an API. It highlights important concepts such as state management, API integration, error handling, and dynamic UI updates. The implementation ensures a smooth user experience and accurate financial data presentation.