Project Overview

This project is a web-based dashboard application designed to visualize and analyze transaction data. The primary objective is to provide users with insights into transaction patterns, such as gender distribution, fraud detection, and spending habits across different merchants and categories. The frontend leverages React and Chart.js for dynamic visualizations, while the backend is powered by Node.js, Express, and MongoDB.

Technology Stack

Frontend:
- **React:** A JavaScript library for building user interfaces.
- **Chart.js:** A charting library used to create various types of charts and graphs.
- **Axios:** A promise-based HTTP client used for making API requests to the backend.

Backend:
- Node.js: JavaScript runtime environment for executing server-side code.
- Express.js: A web application framework for Node.js that simplifies routing and server setup.
- MongoDB: A NoSQL database used for storing transaction data.
- **Mongoose:** An ODM (Object Data Modeling) library for MongoDB, used to define schemas and interact with the database.

Middleware:
- CORS: Middleware to allow cross-origin requests from the frontend to the backend.
- bcrypt:(Planned for future use) A library for hashing passwords, intended for authentication.

Application Features

1. Transaction Data Visualization:
   - The application displays transaction data through pie charts and bar graphs. Users can view insights such as:
     - Gender Distribution: The proportion of transactions made by male and female customers.
     - Fraud Detection: The distribution of fraudulent versus non-fraudulent transactions.
     - Spending Patterns: Total amounts spent categorized by merchants and transaction types.

2. Data Filters:
   - Users can apply filters to analyze specific subsets of the data, such as transactions by gender, fraud status, or specific categories.

3. Authentication (Planned but Not Implemented):

  - The project includes plans for an admin authentication system using bcrypt for password hashing. The goal is to restrict access to the dashboard to authorized users only.

Folder Structure and Architecture

Backend (`/backend`)

- /conn: Handles the connection to the MongoDB database.
- /models: Contains Mongoose schemas, including the transaction schema that defines the structure of transaction data.
- /routes: Contains route handlers that define the API endpoints. For example:
  - transactions.js: Manages routes related to fetching and managing transaction data.

- app.js: The entry point of the backend application. It sets up middleware, routes, and starts the server.

Frontend (`/frontend`)

- /components: Contains React components used for building the user interface. Key components include:
  - /dashboard: Handles data visualization with components like `Dashboard.jsx` and `Calculation.jsx`.
  - /sidebar: Provides navigation within the dashboard.

- /styles: Contains CSS files for styling the application components.

- App.js: Manages the routing and rendering of components within the React application.
- index.js: The entry point of the React application that renders the app into the DOM.

How to Run the Project

1. Install Dependencies:
   - In both the backend and frontend directories, run `npm install` to install the required dependencies.

2. Start the MongoDB Database:
   - Ensure MongoDB is running locally.

3. Run the Backend:
   - In the backend directory, run `node app.js` or `nodemon app.js` to start the server.

4. Run the Frontend:

- In the frontend directory, run `npm start` to start the React application.

5. Access the Application:
   - Navigate to `http://localhost:3000` in your browser to view the dashboard.

Future Enhancements

1. Admin Authentication:
   - The project includes a plan to implement an authentication system using bcrypt for securing the admin login. The login credentials would be stored in MongoDB, and the authentication logic would restrict access to the dashboard to authorized users.

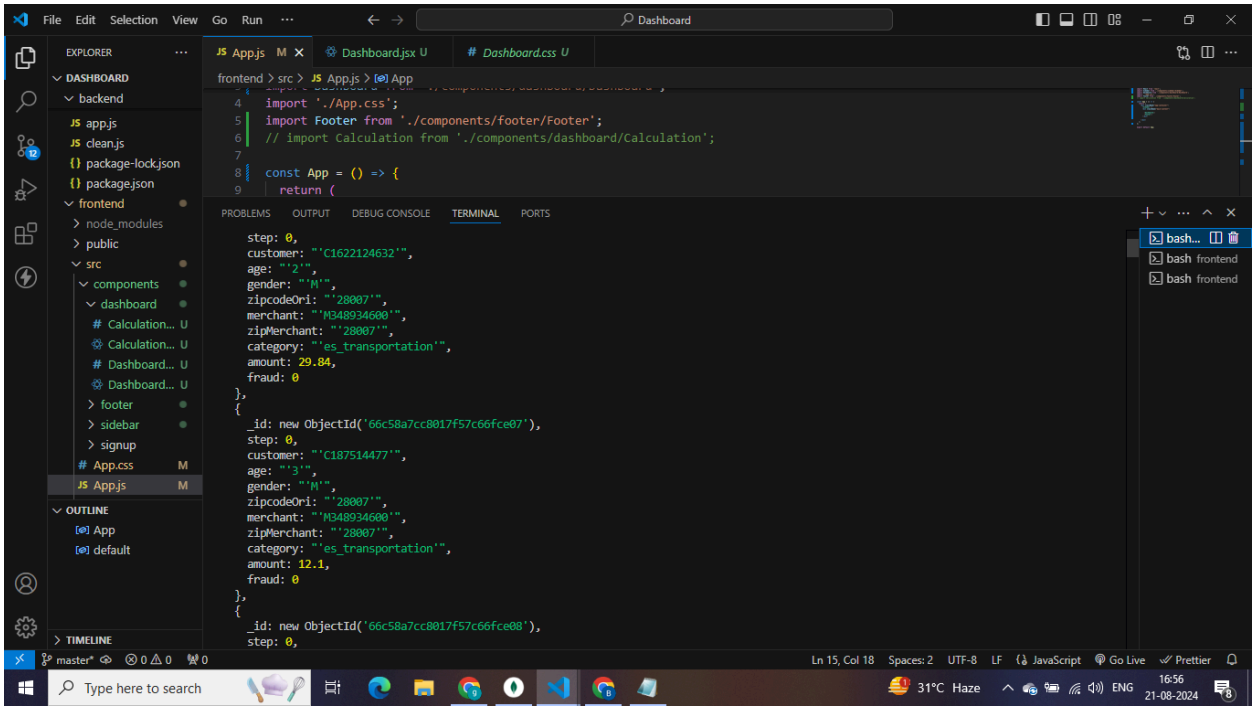
2. Additional Visualizations:
   - The application can be extended to include more complex visualizations, such as time-based trends and comparative analyses.

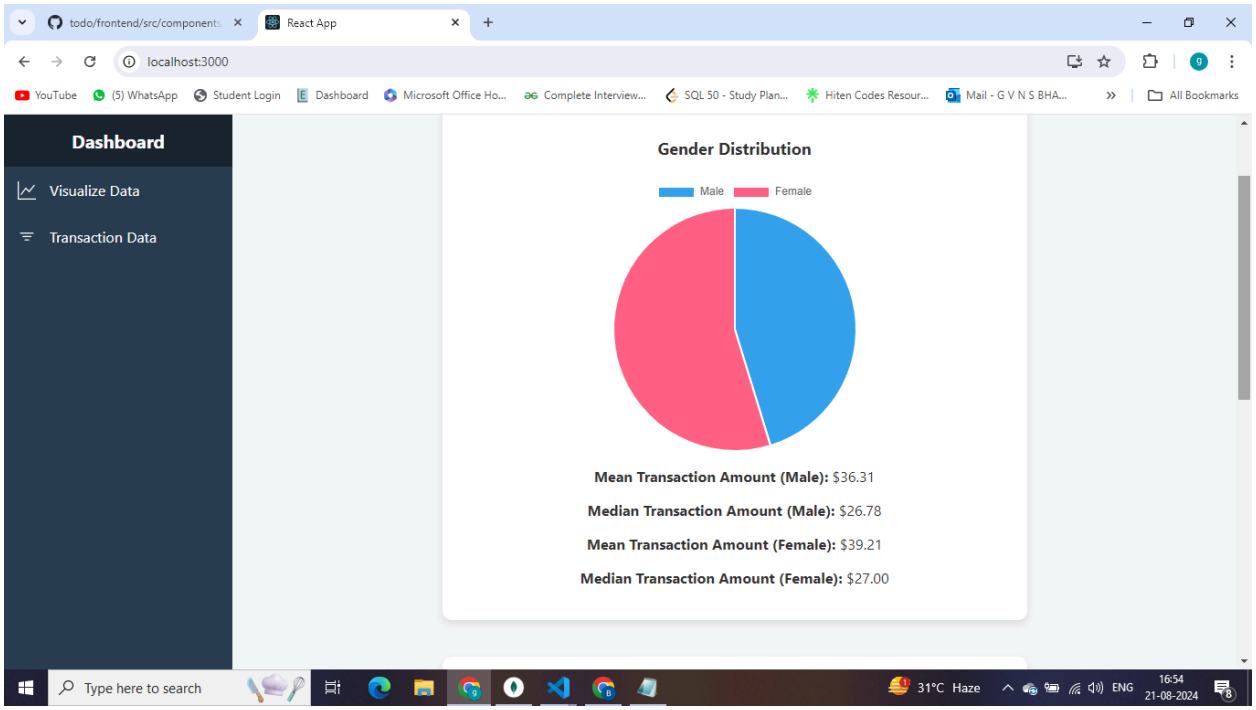
3. Data Export Features:
   - Adding the ability to export filtered data to CSV or Excel formats could further enhance the usability of the dashboard.

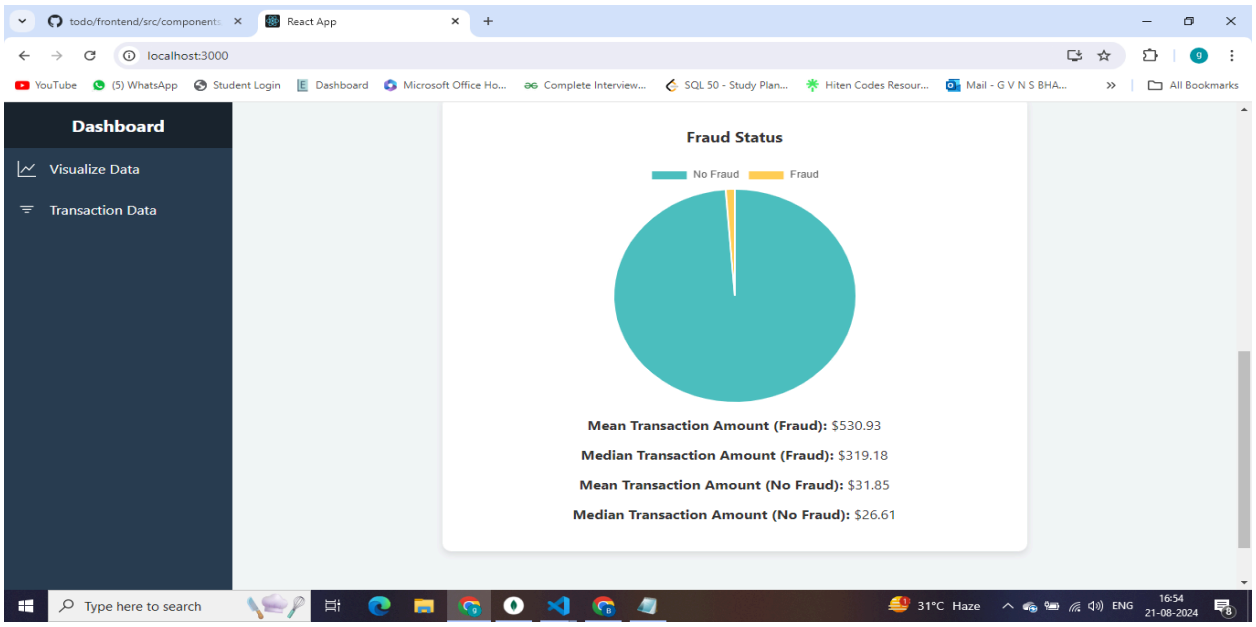
Conclusion

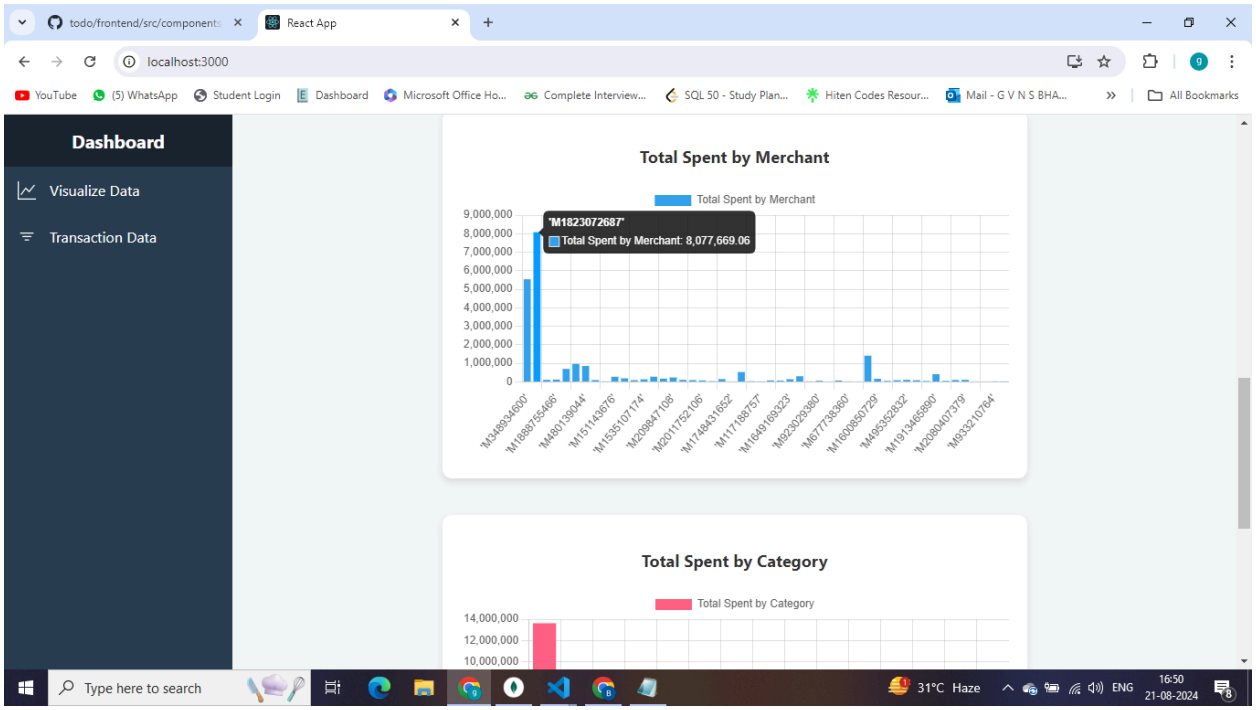
This project is designed to provide valuable insights into transaction data through a user-friendly interface. While the core functionality is in place, including data visualization and filtering, future updates can enhance security and add more advanced features. The architecture is built to be modular and scalable, making it easy to extend the application as needed.

Screenshots


File Edit Selection View Go Run
Dashboard
EXPLORER
DASHBOARD
backend
app.js
clean.js
package-lock.json
package.json
frontend
node_modules
public
src
components
dashboard
Calculation...
Calculation...
Dashboard...
Dashboard...
footer
sidebar
signup
App.css
App.js
OUTLINE
App
default
TIMELINE
App.js M
Dashboard.jsx U
Dashboard.css U
frontend > src > App.js > App
import './App.css';
import Footer from './components/footer/Footer';
// import Calculation from './components/dashboard/Calculation';
const App = () => {
return (
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
bash...
bash frontend
bash frontend
step: 0,
customer: "'C1622124632'",
age: "'2'",
gender: "'M'",
zipcodeOri: "'28007'",
merchant: "'M348934600'",
zipMerchant: "'28007'",
category: "'es_transportation'",
amount: 29.84,
fraud: 0
},
{
_id: new ObjectId('66c58a7cc8017f57c66fce07'),
step: 0,
customer: "'C187514477'",
age: "'3'",
gender: "'M'",
zipcodeOri: "'28007'",
merchant: "'M348934600'",
zipMerchant: "'28007'",
category: "'es_transportation'",
amount: 12.1,
fraud: 0
},
{
_id: new ObjectId('66c58a7cc8017f57c66fce08'),
step: 0,
master*
Ln 15, Col 18 Spaces: 2 UTF-8 LF JavaScript Go Live Prettier
Type here to search
31°C Haze
ENG
16:56
21-08-2024


todo/frontend/src/components
React App
localhost:3000
YouTube
(5) WhatsApp
Student Login
Dashboard
Microsoft Office Ho...
Complete Interview...
SQL 50 - Study Plan...
Hiten Codes Resour...
Mail - G V N S BHA...
All Bookmarks
Dashboard
Visualize Data
Transaction Data
Gender Distribution
Male
Female
Mean Transaction Amount (Male): $36.31
Median Transaction Amount (Male): $26.78
Mean Transaction Amount (Female): $39.21
Median Transaction Amount (Female): $27.00
Type here to search
31°C Haze
ENG
16:54
21-08-2024


todo/frontend/src/components
React App
localhost:3000
YouTube
(5) WhatsApp
Student Login
Dashboard
Microsoft Office Ho...
Complete Interview...
SQL 50 - Study Plan...
Hiten Codes Resour...
Mail - G V N S BHA...
All Bookmarks
Dashboard
Visualize Data
Transaction Data
Fraud Status
No Fraud
Fraud
Mean Transaction Amount (Fraud): $530.93
Median Transaction Amount (Fraud): $319.18
Mean Transaction Amount (No Fraud): $31.85
Median Transaction Amount (No Fraud): $26.61
Type here to search
31°C Haze
ENG
16:54
21-08-2024


todo/frontend/src/components
React App
localhost:3000
YouTube
(5) WhatsApp
Student Login
Dashboard
Microsoft Office Ho...
Complete Interview...
SQL 50 - Study Plan...
Hiten Codes Resour...
Mail - G V N S BHA...
All Bookmarks
Dashboard
Visualize Data
Transaction Data
Total Spent by Merchant
Total Spent by Merchant
'M1823072687'
Total Spent by Merchant: 8,077,669.06
9,000,000
8,000,000
7,000,000
6,000,000
5,000,000
4,000,000
3,000,000
2,000,000
1,000,000
0
'M348934600'
'M1888755466'
'M480139044'
'M1511143676'
'M1535107174'
'M209847108'
'M2011752106'
'M1748431652'
'M117188757'
'M1649169323'
'M923029380'
'M677738360'
'M1600850729'
'M495352832'
'M1913465890'
'M2080407379'
'M933210764'
Total Spent by Category
Total Spent by Category
14,000,000
12,000,000
10,000,000
Type here to search
31°C Haze
ENG
16:50
21-08-2024