MERN TEST - TECHNICAL MANUAL

Project Description

The task is to develop a dashboard application using the MERN stack (MongoDB, Express, React, Node.js) with Apollo Server v4 and GraphQL for data management.

Development Approach

Seperate the task into multiple parts. Beginning from the backend to Setup the **Node JS** and **MongoDB Database connection and Configuration.** Implemented **Express JS** to easy out the Node JS server configurations.

Faker(npm library) is used to insert the fake or dummy data as per requirement into the database with Schema constrain. **Mongoose** (npm Library) is used to manage the Schema definition and other mongoDB operation.

Node environment is consist of **TypeScript** constrain inorder to make development fluent and safe. Elsint is initiated at the time of Node initialization to follow the codding standards.

As per the requirement **GraphQL** is configured in order to handle the API and strict response. GraphQL schema and resolvers are created in way that will follow official guidelines of GraphQL documentation.

Continue with the Frontend: **React js with TypeScript template** is used to construct the Dashboard Ui withCharts. Followed the conventional and recommended Folder structure and Routing methods. **Redux Saga** is implemented to manage the states in react functional components along with Redux, Redux toolkit and Redux thunk. For the dashboard chart project uses the **Nivo Charts** and **Chat js 2.**

For the testing purpose projects uses the **Jest Testing** (npm library). The charts rendering on the Dashboard and Data fetching throughout the project is well tested with the Jest testing.

Tools Used in Development Process

- VS Code
- Postman
- Git (Version Control)
- Appolo Server Sandbox

How to RUN Project (Steps)

(1)
In backend add **.env** file in the root folder and put your mongoDB connection url in following form.

```
MONGO_URL= YOUR_MONGODB_CONTTECTION_URL
```

(2)
Hit following commands in the commad prompt navigated to the project directory.

- **npm install**
- **npm install --global nodemon**
- **npm run dev**
- **npm run test**

(3)
Fake filling Procedure: for filling fake data we have to hit following APIs in given Secuence. (**Use Postman Documentation**)

- **http://localhost:5000/fake/products**
- **http://localhost:5000/fake/users**
- **http://localhost:5000/fake/orders**

(4)
In frontend add **.env** in root folder and put backend server link in the following form:

```
REACT_APP_BACKEND_URL= http://localhost:5000
```

(5)
Navigate to the frontend project. And hit the following commands in the terminal.

- **npm install** or **npm install –force**
- **npm start**
- **npm run test**