**Full Stack Development with MERN**

**Project Documentation – Grocery Web App**

**1. Introduction**

**Project Title:** Grocery Web App

**Team Members**:

MOHAMMED RAFFIYUTHEEN J–Leader

GOPIKRISHNAN A

SANTHOSHKUMAR J

SRINIVASAN K N

**2. Project Overview**

**Purpose**:

A grocery app aims to streamline the shopping experience by enabling users to browse and purchase groceries conveniently from their devices. It helps in saving time and effort with features like home delivery, quick reordering, and real-time stock updates. The app enhances user convenience and promotes efficient shopping through personalized product recommendations and easy payment options.

**Features**:

- User-Friendly Interface

- Order Tracking

- Smart Search & Filters

- Secure Payment Options

**3. Architecture**

**Frontend**:

The frontend is built using React, providing a dynamic and responsive user interface. Components are organized to manage state efficiently and ensure a smooth user experience. Key libraries include React Router for navigation and Axios for handling API requests.

**Backend**:

The backend uses Node.js and Express.js to handle API requests, manage routes, and interact with the MongoDB database. The architecture is modular, with organized routes, controllers, and middleware to streamline backend functionality.

**Database**:

MongoDB is used as the primary database. The database schema includes collections for users, properties, and bookings. Key interactions include creating and fetching user data, Grocery details, and Orders records.

**4. Setup Instructions**

**Prerequisites:**

- Node.js (v14 or higher)

- MongoDB

**Installation**:

1. Clone the repository:

```bash

git clone [repository\_url]

```

2. Navigate to the project directory and install dependencies:

```bash

cd backend

npm install

```

3. Create a `.env` file in the `backend` directory with the following environment variables:

```plaintext

MONGO\_DB=<your\_mongodb\_connection\_string>

PORT=3000

JWT\_SECRET = <your secret key>

BRAINTREE\_MERCHANTID = <your merchant\_id>

BRAINTREE\_PUBLICKEY = <your public\_key>

BRAINTREE\_PRIVATEKEY = <your private\_key>

**5. Folder Structure**

**Client:**

The React frontend (if applicable) would typically have directories like `src/components`, `src/pages`, and `src/assets` to organize reusable UI elements, main pages, and media assets.

**Server:**

- config: Contains the `connect.js` file for MongoDB connection.

- controllers: Manages different business logic functions.

- middlewares: Contains middleware functions (e.g., authentication middleware, if applicable).

- routes: Defines API endpoints for various features (e.g., user, property, booking routes).

- index.js: Main server entry point, where Express app is initialized and database connection is configured.

**6. Running the Application**

**Frontend:**

Run `npm start` in the `client` directory to start the React development server.

**Backend:**

Run `npm start` in the `backend` directory to start the Express server.

**7. API Documentation**

**Endpoints**:

- GET /getallusers - Fetches all registered users.

- POST /handlestatus - Updates status for properties/bookings.

- GET /getallproperties - Retrieves available properties.

- GET /getallbookings - Retrieves booking information.

Each endpoint accepts JSON requests and returns JSON responses. Authentication middleware has been removed for open access.

**8. Authentication**

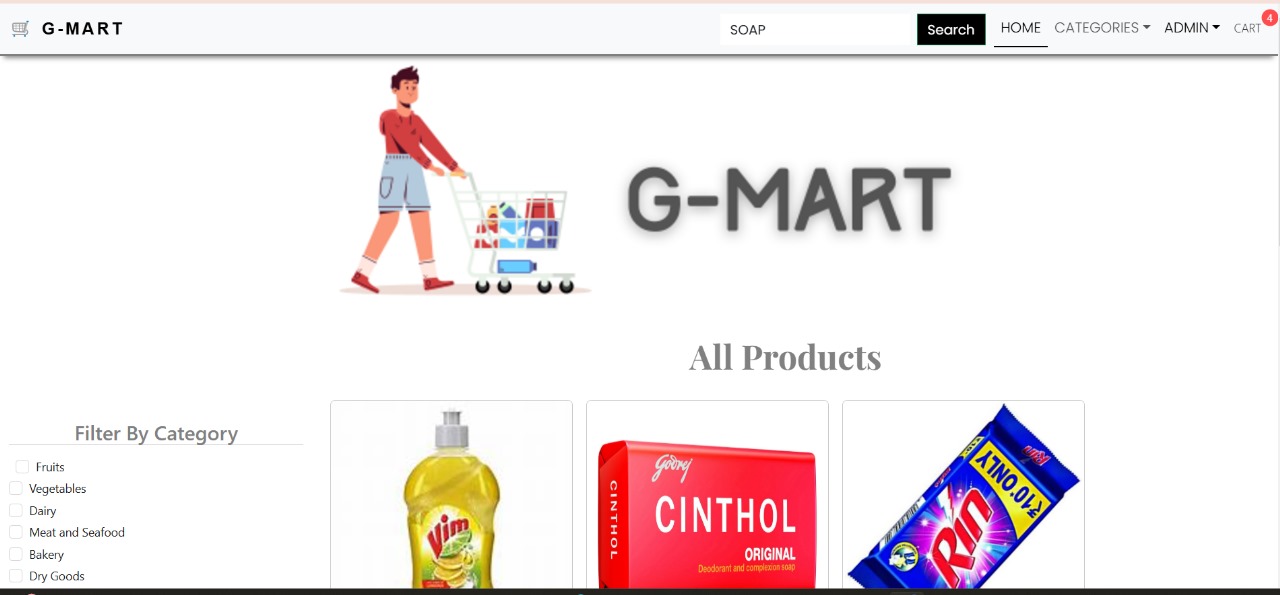
 **API Key Authentication**: Simple token-based authentication where a unique API key is added to the request header.

 **Bearer Token**: Uses an access token (e.g., OAuth 2.0) in the Authorization header for secure, token-based access.

 **Basic Authentication**: Sends a Base64-encoded string of the username and password in the request header.

 **OAuth 2.0**: A robust method that enables token-based authorization, involving access tokens and refresh tokens for secure, scalable authentication.

**9. User Interface**



**Overview**:

A grocery app enhances the shopping process by allowing users to browse, purchase, and manage groceries with ease from their devices. It offers convenient features like personalized recommendations, secure payments, and real-time order tracking. This app streamlines shopping, making it faster and more accessible.

**10. Testing**

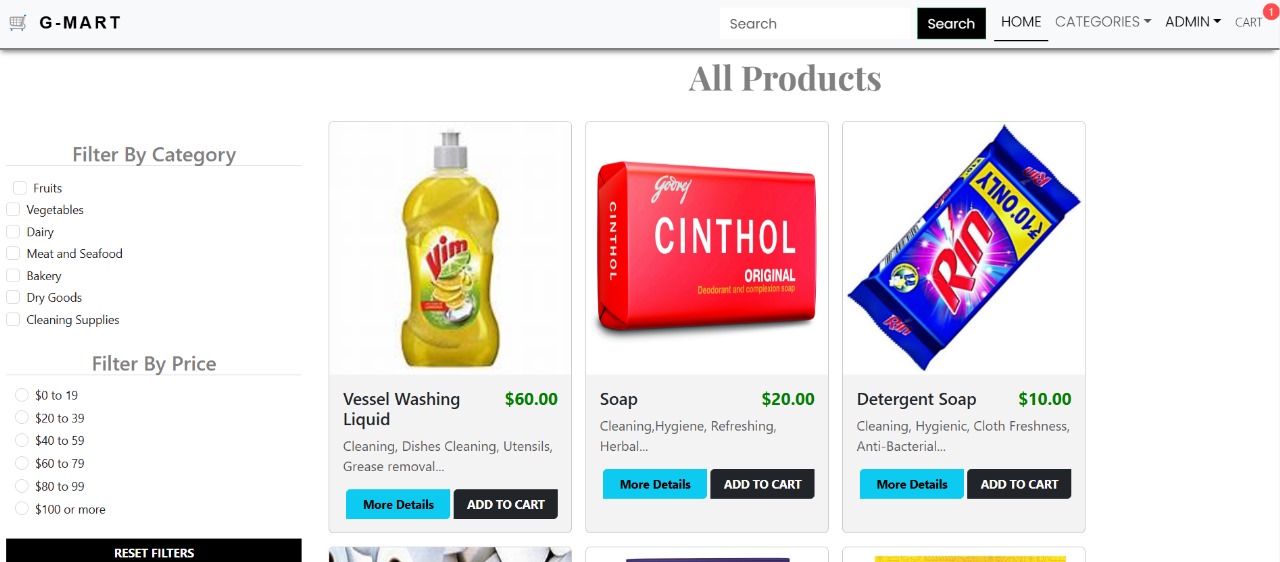
**Testing Strategy**:

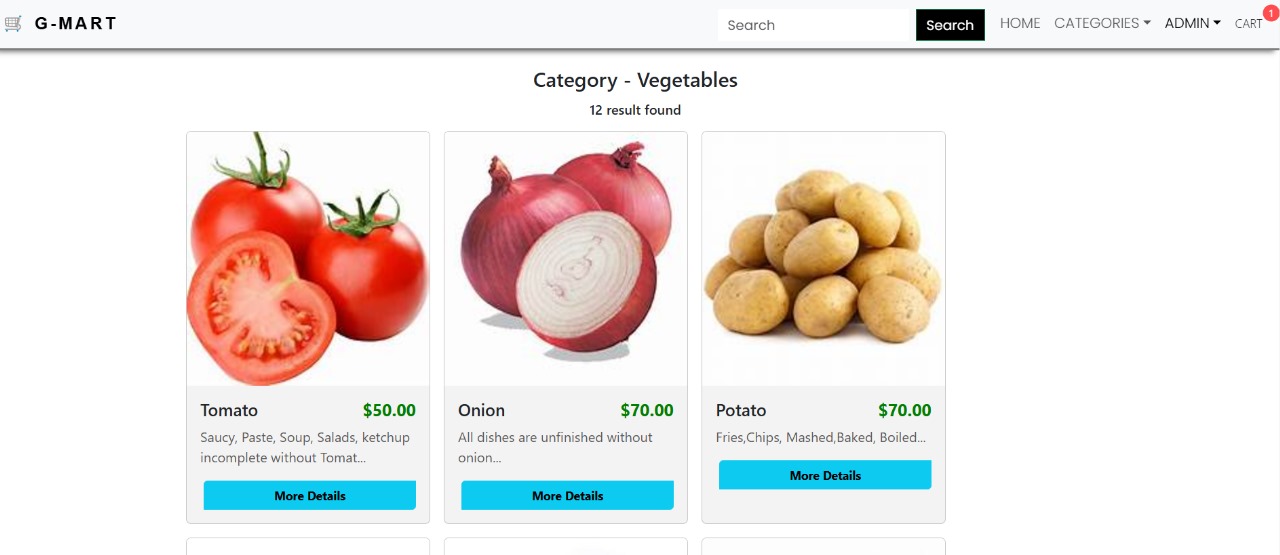
- Manual testing of all frontend and backend functionality.

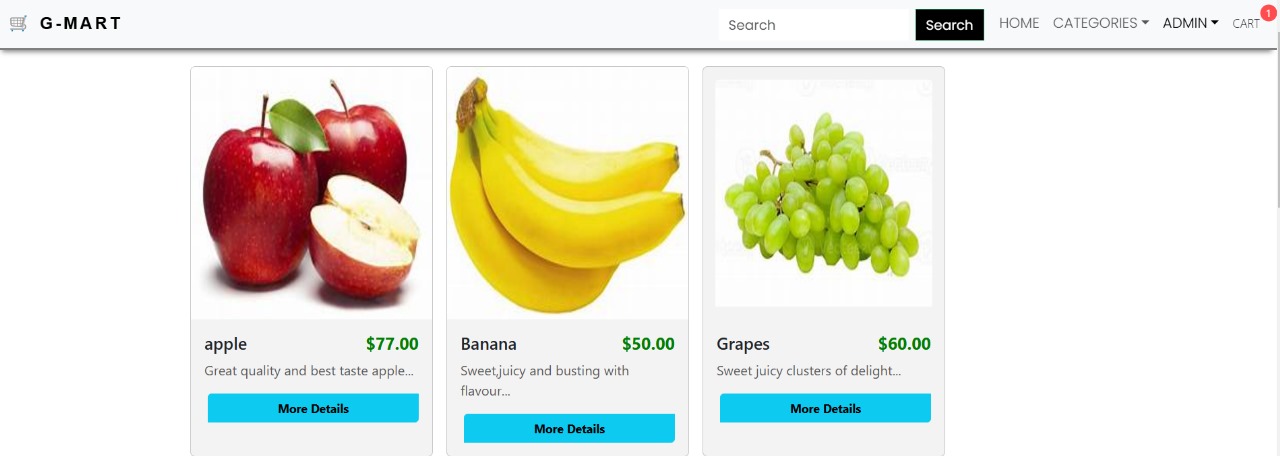
- Postman is used for API testing.

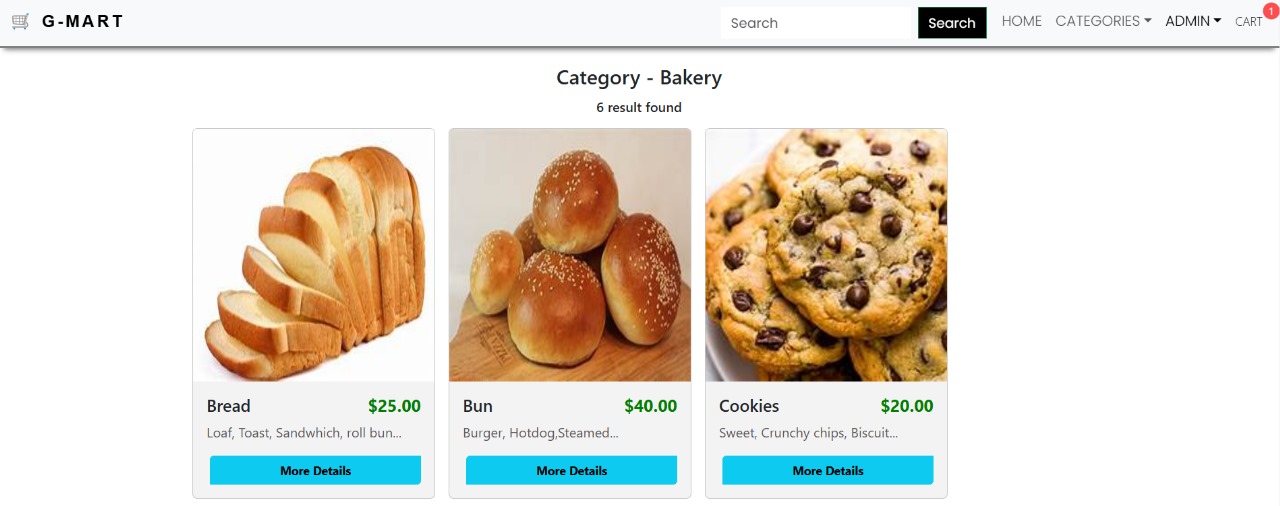
**11. Screenshots or Demo**

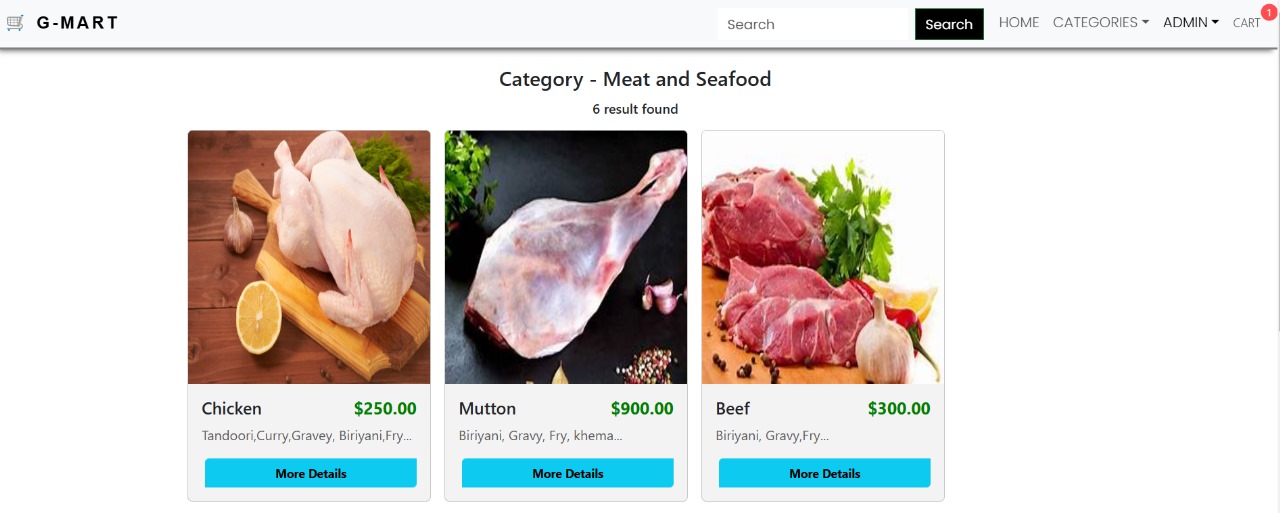
**Screenshots:**

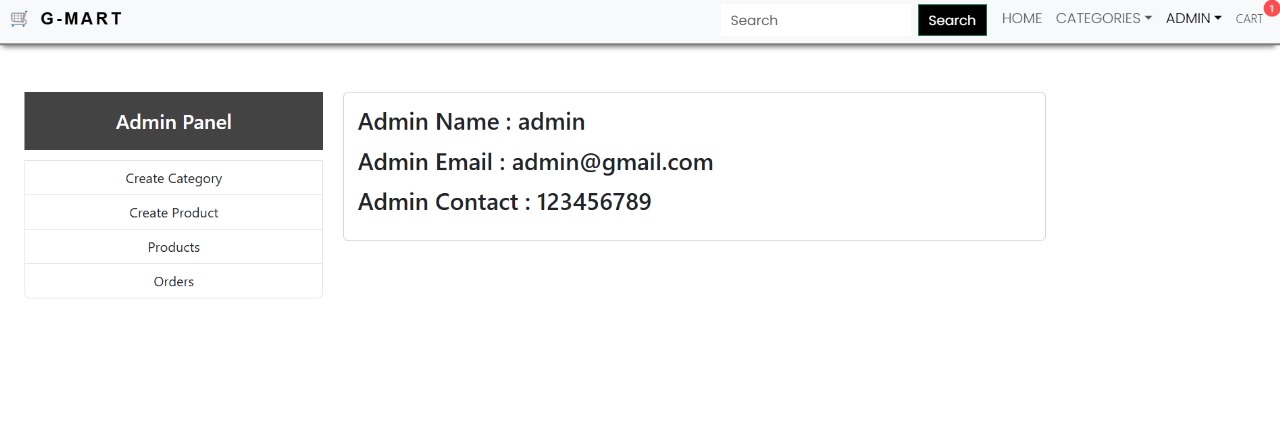


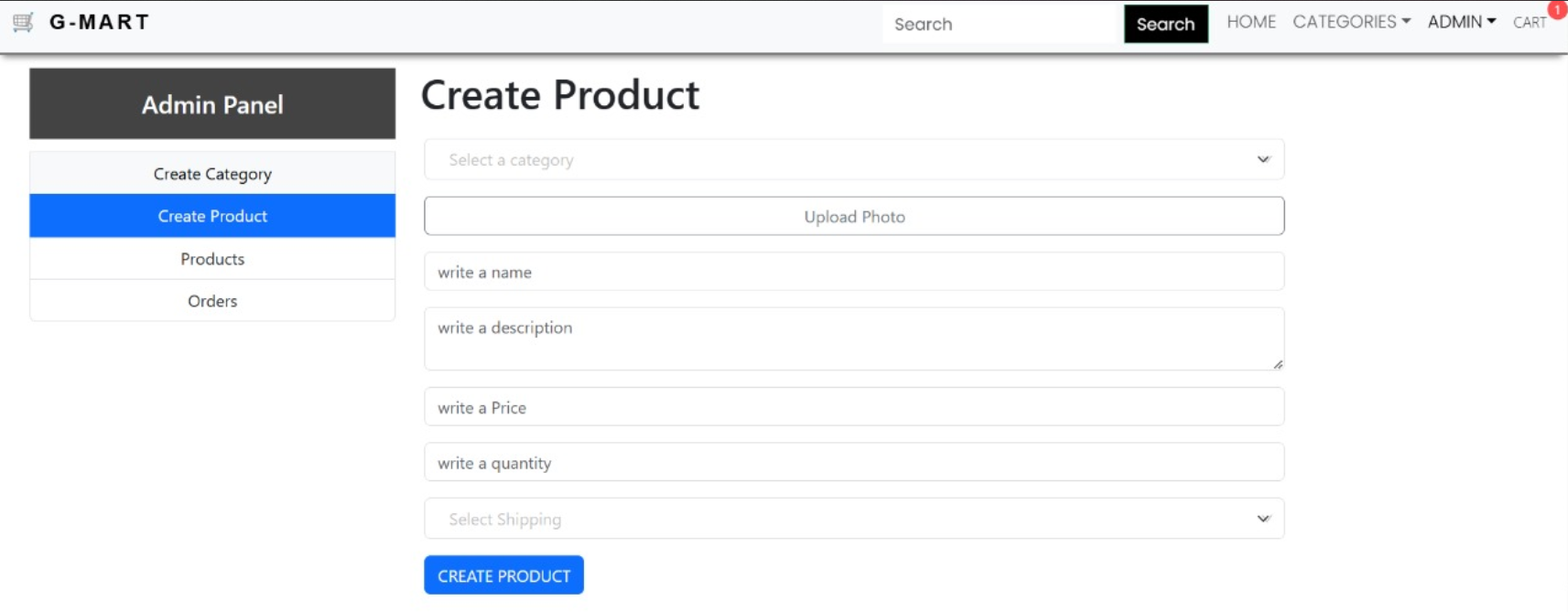


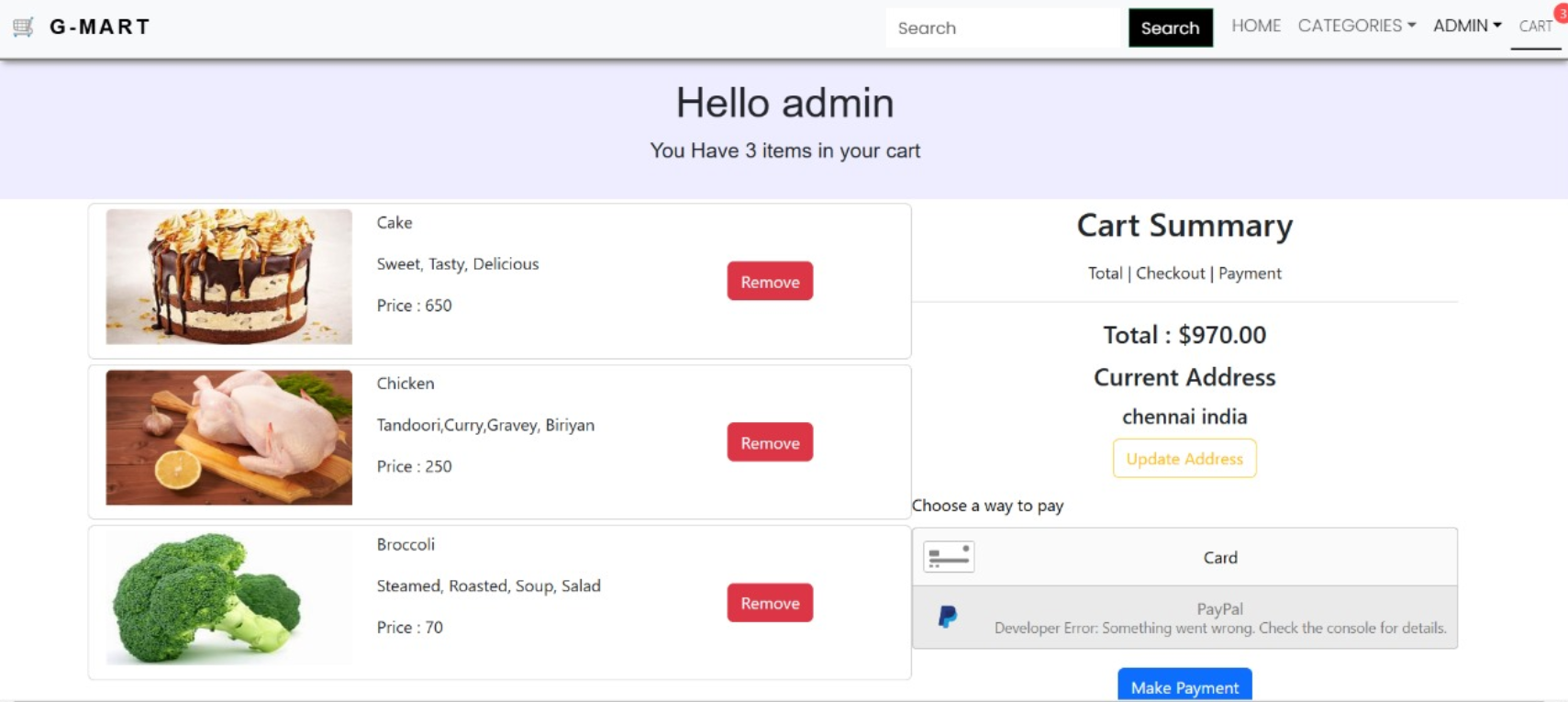


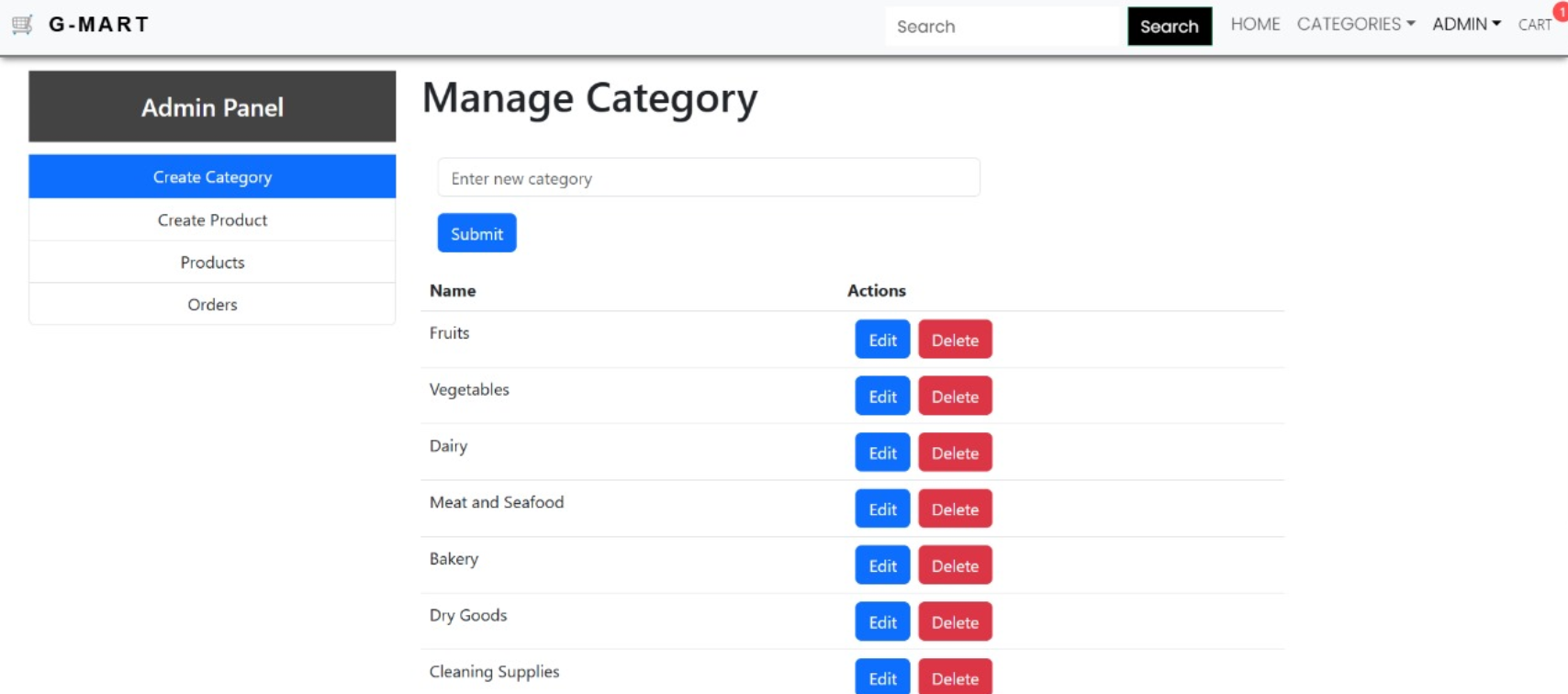


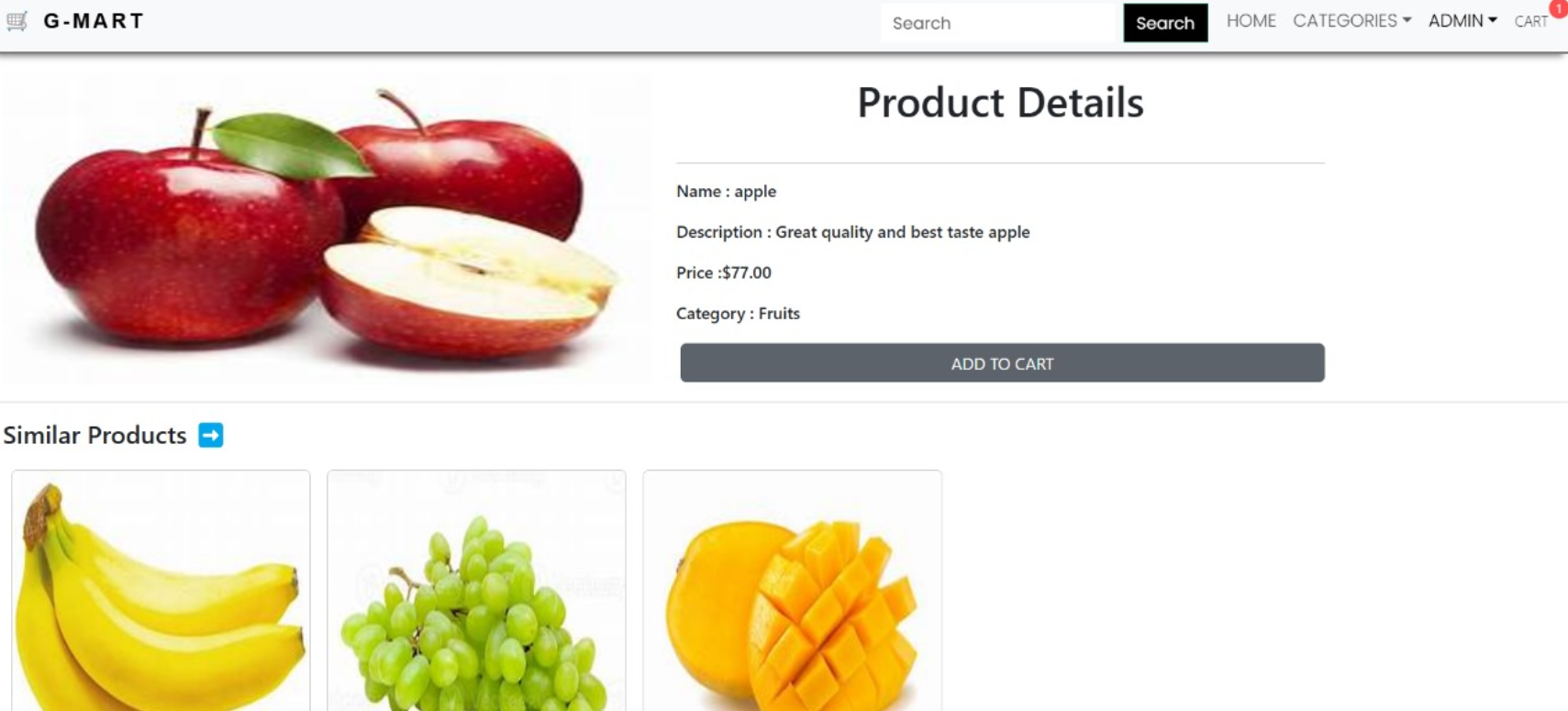


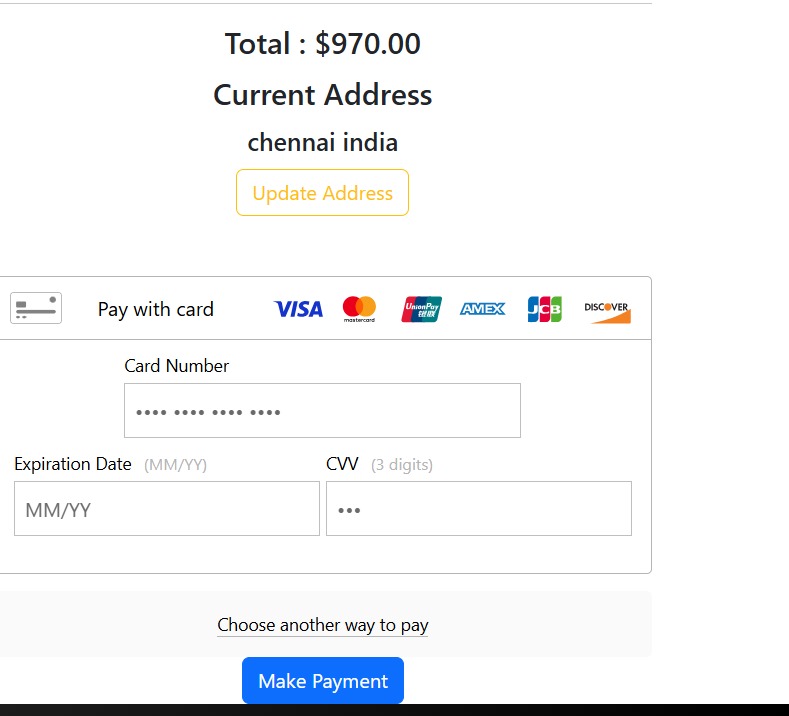


****

****

****

****



**12. Known Issues**

**Issues:**

- Page reload required after order placed .

**13. Future Enhancements**

 **Personalized Shopping Experience**: Uses AI to suggest products based on user preferences, past purchases, and browsing history.

 **Subscription and Auto-Reorder**: Allows users to set up recurring orders for frequently bought items to save time.

 **In-App Chat Support**: Offers real-time customer service for inquiries, complaints, and assistance.

 **Barcode Scanner**: Lets users quickly add items to their cart by scanning product barcodes at home.

 **Nutrition and Allergen Information**: Displays detailed product information, including nutritional values and potential allergens, for health-conscious shoppers.

 **Discount Alerts and Loyalty Rewards**: Provides notifications about special deals and rewards points to retain customer engagement.