MAD Project II - Grocery Store Web Application

Name - Rekha Project-present-video-link Roll No. - 21f1006795

Email Id - 21f1006795@ds.study.iitm.ac.in

Overview

The Grocery Store Web Application is a comprehensive solution designed to streamline and enhance the operations of a grocery store. The application facilitates the management of products, orders, and user interactions, providing a seamless experience for customers, store managers, and administrators.

System Components

1. User Management

- User: Can browse products, add items to the cart, and place orders.
- Store Manager: Manages product listings, requests CRUD on categories, and generates reports.
- Admin: Oversees user and manager registrations, manages categories..

2. Product and Category Management

- Products are organized into categories.
- Managers can add, edit, and delete products and request add, edit and delete categories and Admin can approve or reject requests..
- Users can search for products and view details.

3. Cart, Checkout, Orders

- Users can add/remove and update a quantity of items from the cart.
- Order confirmation and all orders with total Expenditure in Orders Section.

4. Reporting - Store Managers can generate reports, including product details, sales, and user activity.

Models

1. User Model - Fields: User ID**,** First Name, Last Name, Username, Password, Email, Role(User, Manager and Admin).

2. Product Model - Fields: Product ID, Product Name, Price, Unit, Expiry Date, Quantity, Image Path, Category, User ID, Role, created_at and updated_at.

3. Category Model - Fields: Category ID and Category Name.

4. Order Model - Fields: Order ID, User ID Role, , Product ID, Product Name, Price, Unit, Product Image, Quantity, Total Price, created_at and updated_at.

System Design

1. Frontend - Developed using Vue.js for a dynamic and responsive user interface. Vuex state for state management and Bootstrap for CSS.

2. Backend - Built with Flask, Python web framework. SQLite database for data storage. Celery and Redis for asynchronous task processing. Flask-cache for caching.

3. Authentication - JWT (JSON Web Token) for secure user authentication and authorization. Role-based access control for different user types.

Technologies Used

- Frontend: Vue.js, Vuex(state management)
- Backend: Flask (Python), SQLite(Database), JWT for secure authentication. Celery for background task execution(like sending monthly reports and export csv report of product sales)