The Fresh Corner

Modern Application Development 1 project

Author:-
Rohit Prajapat
22f1001536
22f1001536@ds.study.iitm.ac.in
About me : A proud student of IITM and a passionate developer. I'm from Rajasthan and I love programming, cycling and reading books.

Description:-

The "**Grocery Store**" project is a web application designed for modern shopping experiences. Built using Flask, Jinja2 templates, and SQLite, it provides a user-friendly interface for customers to browse and purchase grocery products, while also empowering store managers to efficiently manage inventory.

Technologies used:-

1. Flask - As the main backend web framework.
2. Sqlite - Data query and schema design
3. Flask-Sqlalchemy - For easy management and integration of databases in flask app.
4. Flask-Login - For handling user authentication efficiently and securely.
5. Werkzeug-Security - For generating password hash.
6. Flask-migrate - For easy migration and management of databases.
7. Flask-restful - For api creation.
8. Python requests library - For calling and fetching data.
9. HTML, CSS, Jinja2 and Bootstrap - For front end development.
10. Swagger/yaml - for api documentation.

DB Schema Design:-

- **User Table:** Stores user details with columns for ID, name, username, password, admin status, and purchase history. User ID is a primary key, username is unique, and password is securely stored using hashing.

- **Product Table:** Contains product information like ID, name, MFD, EXP, rate, unit, description, image, mimetype, quantity, category ID, and deletion status. Product ID is the primary key, image is stored as a binary text, and the category ID is a foreign key linked to the Category table.

- **Category Table:** Holds category details with columns for ID, name, and deletion status. Category ID is the primary key, and the name is unique.

- **PurchaseHistory Table:** Tracks purchase details, including ID, purchase date, user ID, product ID, and quantity. Purchase ID is the primary key, and user ID and product ID are foreign keys.

API Design:

- Created API endpoints for managing products and categories in a grocery store.
- Implemented using Flask-RESTful library.
- Endpoints include: GET, PUT, DELETE for products/categories and GET for lists.
- Utilized request parsers for data validation and response fields for structured output.
- Implemented user authentication for certain admin-only endpoints.
- Designed API resources for Product and Category, with appropriate methods and field marshaling.
- Achieved clear separation of concerns, allowing for smooth interaction between frontend and backend.

Project Layout:-

Grocery-Store

```
├── apis
│   ├── api.py
│   └── __init__.py
├── api_documentation.yml
├── app.py
├── database
│   └── store.db
├── FreshCorner # main app
│   ├── admin.py
│   ├── authentication.py
│   ├── cart.py
│   ├── model.py
│   ├── public.py
│   ├── user.py
│   ├── validation.py
│   └── __init__.py
├── migrations
│   ├── alembic.ini
│   ├── env.py
│   ├── README
│   ├── script.py.mako
│   └── versions/
├── README.md
├── requirements.txt
├── static
│   ├── Images/
│   │
│   ├── product_img/
│   │
│   └── style.css
└── templates
    ├── admin
    │   ├── addproduct.html
    │   ├── add_category.html
    │   ├── admin.html
    │   ├── category_page.html
    │   ├── delete_category.html
    │   ├── delete_product.html
    │   ├── products.html
    │   ├── update_category.html
    │   └── update_product.html
    ├── base.html
    ├── cart.html
    ├── categories.html
    ├── category_page.html
    ├── dashboard.html
    ├── index.html
    ├── insights.html
    ├── login_user.html
    ├── products.html
    ├── product_detail.html
    ├── register.html
    └── search_results.html
```

Note: Api and app are running simultaneously in the create_app function in app.py i.e. outside the app. Also both are running on the same url.

Video url :- https://youtu.be/28Em137m3NY