Grocery Store

Author

Name: Hitesh Binjrawat
Roll No: 22f2001255
Email ID: 22f2001255@ds.study.iitm.ac.in
I am pursuing my Bachelor's Degree from Indian Institute of Technology, Madras in Data Science & Applications. Also pursuing BSc Computer Science from University of Delhi. I am passionate about machines and robots.

Project Description

Grocery Store is an online platform for buying groceries. It has two types of members - User & Admin, who can register themselves. Users can search for products, buy products and create carts. Admins can create and delete categories and products.

Technologies Used

1. Flask - Used for application code.
2. Jinja2 templates, CSS and Bootstrap - For HTML generation and styling.
3. SQLite and SQLAlchemy for Data Storage and Querying through data.

DB Schema Design

The database has 4 tables:

1. **user:** user_id, full_name, email, username, password, role.
   The table **user** contains the information about the registered users which are identified by the roles differently.

2. **Section:** category_id, category_name, admin_id(Foreign Key("user.user_id")).
   The table **category** contains the information about different categories of products added. It contains a foreign key 'admin_id' which relates the category to the admin who created the category.

3. **Products:** product_id, product_name, unit, rate_per_unit, mfg_date, exp_date, total_quantity, category_id(Foreign Key("category.category_id")).
   The table **products** contain the data of all the products added. This contains a Foreign key 'category_id' which relates it to the category which it belongs to.

4. **Order:** order_id, user_id(Foreign Key("user.user_id")), product_id(Foreign Key("products.product_id")),admin_id, Quantity, price, is_Active, Date, transaction_id .
   The table **orders** contain the data of all the orders that were placed by any user. This contains many foreign keys like user_id - for the user who ordered the product, product_id - for the product that was ordered. The orders where is_Active – "No" refer to those which

were placed for order, whereas is_active – “Yes” refers to those which were placed in the cart.

Architecture and Features

The project is divided into many subfolders and files - application, db_directory, static, templates, app.py,
The directory - **application** contains all the python files required to run the application.
All the html files to render are stored in the **templates** folders.
The **db_directory** directory stores the database for the application namely database.sqlite3.
The **static** folder contains all the other items like images used in the application.
The file **app.py** is the main file for hosting the application.

When the application is executed, the member can login according to the role - Admin or User. The application also has a create new user feature also for the new users.
For the Admin section, the create, read, update & delete (CRUD) feature is applicable for both categories and products.
For the User section, the user can buy or add the product to the cart using different buttons for different actions. The cart shows the products added in the cart it can contain single as well as multiple products. The user can also delete the product from the cart.
The user also has the option to search for the product where the product can be purchased or added to cart. The user can look his orders in the section myorders inside the navigation bar.

Video

https://drive.google.com/file/d/1ip_l6YPKu-ssraiy1_NwWzvFnrR2o9Ge/view?usp=sharing