Author

Name: Nivedhaa Srikanth

Roll No: 21f2000953

Student Mail: 21f2000953@ds.study.iitm.ac.in

B.sc Physics Graduate, a motivated learner, excited to learn something new every day. I am also a musician and tech and space enthusiast.

Description

The purpose of this project is to design and implement a fully functional grocery store app using the Flask framework, Vuejs and an SQLite database. The primary objective is to create a fully functional platform that seamlessly connects customers with products by creating a user-friendly interface while enabling effective inventory management for administrators and store managers. Scheduled jobs have been implemented which ensures that users remain engaged. Additionally, another scheduled job has been implemented which generates monthly reports that track sales trends, offering valuable insights into the app's performance and customer behaviour. Store managers also can export product details as CSV files, which can be used for further analysis.

Technologies used

1. Flask Framework: The core of the application is built upon the Flask web framework.

2. Flask-RESTful: The implementation of Flask-RESTful has enabled the development of RESTful API endpoints to perform CRUD operations efficiently.

3. Vue js: For Frontend interface.

4. Flask-Security: RBAC login has been implemented making use of flask_security authentication.

5. Flask-Bcrypt: To enhance the security of user data by proficiently hashing passwords before they are stored in the database.

6. SQLite Database: The choice of SQLite as the database management system has played a

pivotal role in storing and managing crucial application data. This technology has underpinned the storage of product details, user information, categories, and cart data.

7. flask_sqlalchemy - for easy integration of SQLAlchemy, a powerful ORM library, into the Flask applications.

8. Werkzeug: For handling HTTP exceptions.

9. JSON: A lightweight data-interchange format which was used to serialize and deserialize data in the application, making it easy to pass data to and receive data from the API.

DB Schema Design

The database schema comprises several tables, each representing a specific entity within the application. The tables are interconnected through relationships, facilitating the storage and retrieval of data in a structured manner.

1. User Table
2. Category Table
3. Product Table
4. Cart Table
5. Order Table
6. category_request
7. Role
8. roles_users

API Design

I have created API endpoints for carts, sections, products and other end-poins that enable users to perform CRUD (create, read, update, delete) operations on each of these entities efficiently. Most of the endpoints look for a get/post request on an associated form, take in the item ID as a parameter and make a request to the database for the associated operation. There's Order table which can be used to see orders made by customers at time.

Architectures and Features

The project is organized into the following directories:

- app.py: This is the main Flask application file.
- application: Contains all the backend files- models.py (tables created), database.py (initialization of database), api.py (Api-end points), task.py
- front-end: Contains all the frontend files- Vue components, routes, App.vue, main.js
- database.db: The SQLite database with all the tables required for this project.

Video: