Author

Shaurya Yamdagni
21f1006750
21f1006750@ds.study.iitm.ac.in
I'm Shaurya Yamdagni, currently pursuing my bachelors in Computer Science and Engineering from SRM University. I am very passionate about learning new things and Implementing new ideas.

Description

Here we need to develop a web app with multiple customers and managers .
The manager can decide all the categories and products with the frontend provided.
The customer can place different orders from different categories.

Technologies used

The main technologies used are Flask for creating the flask application, flask_login for maintaining user sessions and logging in and logging out. Flask_sqlalchemy is used for connecting to the SQLite database. Flask-restful for creating the simple rest API.

DB Schema Design

The database used in the the application is SQLite since it does not require separate server to run and the content can be saved in a single file. The database has 7 tables in the file in The file is named database.sqlite3 .
The Tables used are -

1. Managers - To store the login credentials of the manager (password is encrypted)
2. Customers - To store the login login credentials of the customers (password is encrypted)
3. Category - To store the categories (managed by the manager)
4. Product - To store all the products (categoryId is the foreign key)
5. customerCart - To store the cart items of the customers with customerId as foreign key
6. customerOrders - To store customer orders(customerId is the foreign key)
7. soldProductsData - To store the data for visualisation purpose for the manager
   Link to database schema -> print.pdf

API Design

The API is implemented using the flask-restful
It has 4 urls -

1. ManagerProductsResource - /api/manager/products
    - To perform GET and POST on products
2. ManagerProductResource - /api/manager/products/<int:product_id>
    - To perform GET, PUT and DELETE on product based on productId
3. ManagerCategoriesResource - /api/manager/categories
    - To perform GET and POST on the categories

4. ManagerCategoryResource - /api/manager/categories/<int:category_id>
    - To perform GET, PUT and DELETE based on categoryId
5. ManagerCategoriesProductResource- /api/manager/categoriesProduct/<int:categoryId>/<int:productId>
    - To perform GET request based on productId and categoryId

Architecture and Features

The project is organised in 5 different directories and one main.py file which contains the driver code for the server .

The architecture is as follows :

1. "application" Directory :

   This directory contains -
    1. Controllers - It contains all the routes and endpoints mapped to functions
    2. Models - It contains all the Database tables as classes .
    3. API - It contains the API classes with all the methods defined and mapped in YAML file
    4. Database - It starts the database engine
2. "templates" Directory

   This directory contains all the necessary templates required by Jinja2 to convert to HTML pages.
3. "static" Directory

   This directory contains all the static images which are overwritten my matplotlib to create new graphs
4. "db_directory " Directory

   This directory contains the actual database of the application which is SQLITE3 in our case

Features :

- [ ] Manager
    - [ ] CRUD on products and categories
    - [ ] Get the graphs on top selling products and most active users
- [ ] Customer
    - [ ] Login or create account
    - [ ] View the latest products on home screen and navigate the categories also
    - [ ] Add products to carts and delete products from carts
    - [ ] Place orders for different products

The site is live here - GroceryStore and here too (development server). (hosted on Ubuntu VPS)

Video

Drive
Youtube