Author

Pratyay Piyush Mishra
21f1004971
21f1004971@ds.study.iitm.ac.in

Description

An app to buy groceries online, with user interfaces for backend store managers as well.
Products can fall in different categories, and store managers can use the app to update stocks as well as create new products.

Technologies used

Flask - used as backend server for api
Flask-JWT-Extended - used to authorize users using jwt based tokens
Flask-sqlalchemy - used as database engine to store app data
Celery - used to provide asynchronous task queues to send order summaries, reminders etc to users asynchronously
Flask-Caching - used to cache the images api results in a redis database rather than read them every time from the sqlalchemy database
Jinja2 - used to generate html for mails to customers from a template

DB Schema Design

User model :
Fields - mail,password,salt,role,last_visited,manager_approved
Password stored after hashing. Random salt is used to hash the password stored in the salt column.
Manager_approved column is read only for users with role manager, and is true if manager is approved by admin

Category model:
Fields - name, description, products
Products column is actually a relationship between the Product model and category model with category id as a foreign key in the products table

Approvals model:
Used to store pending category approvals from store managers
Fields - cid,task,name,description

Cid is the category id, and specifies the category on which the store manager is requesting specified action. Is null for requesting addition of a new category
Task is the task type, whether it is a add, delete or edit request
Name and description contain the new name and description of category in case of addition and editing tasks

Products model:
Fields - name, expiry,rate,quantity,catID,units,image,added
catID stores which category the product belongs to, units - number of units in stock, added - the date on which the product was added to database

Orders model:
Used to store orders placed from different users.

Images model - used to store images with ids back referenced to the product to which it belongs

Architecture and Features

Models.py contains the database model classes along with their associated methods to create, read, update and delete
Routes.py contains all the various api endpoints
The celery tasks along with app configuration is in the app_instance.py file
The templates folder contains the html template to generate html mails to be sent to users

All core features implemented, including role-based access control, customer shopping page, manager and admin dashboards with appropriate functionality.
Celery webhooks used to send reminders to customers, and celery tasks to send mails with order summary to customers
Celery tasks used to also send product stock csv to store manager mail id, the task gets triggered on clicking export in manager dashboard.
Redis caching used to store images api results in memory rather than reading every time from database

Video

https://drive.google.com/file/d/14BWdbljaErgzQcgqXrWcrj3UINdpJ4sC/view?usp=sharing