GroceryStore - Modern Application Development – I

Project Report

By

Govind M J (21f1004297)

21f1004297@ds.study.iitm.ac.in

Overview

The project GroceryStore is a web application for managing products, user authentication, shopping carts, and various functionalities. The application is designed using the Flask web framework and employs Jinja2 templates for rendering dynamic HTML content. Below is an overview of the models and the overall system design.

Models

1. User Model: The User model represents registered users of the application. It includes fields such as username, email, password hash, and an 'is_admin' flag to determine admin privileges.
2. Product Model: The Product model represents individual products in the system. It includes attributes like name, manufacture date, expiry date, rate per unit, category, and available units.
3. Category Model: The Category model defines product categories. It includes a simple 'name' attribute.
4. Cart Item Model: The Cart Item model is used to store items added to the shopping cart. It relates to the User and Product models through foreign keys, representing the user who added the item and the product being added.

System Design

The application follows a modular design to ensure separation of concerns and maintainability. Here's an overview of the main components:

1. Templates: Templates are used for rendering dynamic HTML content. They are written in Jinja2 and are organized based on different views like user authentication, product management, shopping cart, and more.
2. Routes: Routes define the URL endpoints for different functionalities. They handle user requests, interact with the models, and render the appropriate templates.
3. User Authentication: The application provides user registration, login, and admin login functionalities. User passwords are securely hashed before being stored.
4. Product Management: Admins can manage products and categories. They can add, edit, and delete products and categories.
5. Shopping Cart: Users can add products to their shopping cart, view the cart's contents, and clear the cart. The cart functionality is associated with the Cart Item model.
6. Search Functionality: The application includes a search feature that allows users to filter products based on category, price range, and date range.

Key Features

- User Registration and Authentication: Users can create accounts and log in securely with passwords for personalized access.
- Admin Login for Product and Category Management: Administrators log in separately to manage products and categories efficiently.

- Product and Category CRUD Operations: Admins can create, read, update, and delete products and categories.
- Shopping Cart Functionality with Cart Items Associated with Users: Authenticated users can add products to their carts, which are linked to their accounts.
- Search Functionality to Filter Products: Users can search and filter products based on various attributes like category, price range, and date.
- Template-Based UI for Different Views: HTML templates dynamically display data, ensuring consistent and organized UI across the app.
- Use of Flask Extensions for Form Handling and URL Routing: Flask extensions simplify form creation/validation and manage URL routing for efficient development.

Conclusion

In conclusion, the GroceryStore project demonstrates a well-structured and feature-rich web application built using the Flask framework. By employing modular design, secure user authentication, efficient product and category management, seamless shopping cart functionality, and effective search capabilities, the application offers a user-friendly and organized platform for online shopping. Through the integration of Flask extensions, the project showcases streamlined form handling and URL routing, contributing to its robust and user-centric design.

Video Link

https://drive.google.com/file/d/1BZTsKiz3vZ_jFNIg8J8DojC6UH-sAbIw/view?usp=sharing