Grocery Store - Modern Application Development – I

Project Report

Sanchay Mahato

21f1002495@ds.study.iitm.ac.in

Video Presentation link:
https://drive.google.com/drive/folders/1LHFdoJw8P7MBzpg4hFS7iwdrOouAS9W4?usp=sharing

app.py

This contains the main Flask application code and routes.

Initialised app and database.

Added models User, Category, Product, CartItem.

Added relationships between models like User-CartItem, Category-Product etc.

Added routes for user registration, login, and logout. This allows users to create accounts, log in, and log out.

Added separate login forms for managers and regular users. Managers can access the manager dashboard, while regular users can only access the user dashboard.

Implemented the manager and user dashboards to display categories and products. Added search functionality in the user dashboard.

Added routes for CRUD operations on categories and products only accessible to logged-in managers.

Added cart and checkout functionality for users to buy products. Users can view the cart, modify cart items, and checkout to place an order.

models.py

Added Order and OrderItem models to store orders placed by users.

The models represent the core entities required for an e-commerce site.

The relationships allow navigation across models, like getting all orders for a user or fetching all products in an order.

forms.py

Added forms for user registration, login, and adding categories/products.

Added a search form for searching products in the user dashboard.

The application uses various forms to manage user input and ensure data validation.

These forms include LoginForm, ManagerLoginForm, UserLoginForm, CategoryForm, ProductForm, SearchForm, RegistrationForm, and UserSearchForm.

They handle login, category and product management, search functionality, and user registration, making the user interactions smooth and secure.

The forms encapsulate the input fields required for different entities and operations like adding products, user registration etc. Validation and sanitisation are handled by WTForms automatically.

These forms handle validation and data sanitisation automatically, ensuring smooth user interactions and enhancing application security.

Summary

The app.py file contains the main Flask application and routes. The routes are defined for core functionality like home page, user registration, login/logout, separate dashboards for managers and users, CRUD operations on products and categories, cart management, checkout and search.

Key routes like add/edit products or categories are restricted to logged-in managers only. Routes use validation functions to sanitise form data before committing to the database.

The user dashboard route implements searching and filtering of products by category and price range. Cart routes handle adding, viewing, modifying and emptying carts for logged-in users. Checkout validates cart items, reduces inventory, clears cart and saves orders on success.

Custom functions are defined for reusable tasks like getting cart items, adding to the cart, clearing cart, searching etc.

The models.py module defines SQLAlchemy models to represent core entities like User, Category, Product, CartItem, Order and OrderItem. Fundamental relationships are defined between models like User-CartItem, Category-Product etc.

The forms.py module defines WTForms classes for all forms used. Forms encapsulate fields, data validation and sanitisation for inputs like user registration, adding products etc.

The app handles user roles, CRUD operations, cart management, and checkout flow. Custom routes, models, forms and helper functions implement the logic and integrate the components. Database relationships allow easy navigation across models. Forms handle validation. Overall, these modules build out a working e-commerce web application.