System Design Report

1. Introduction

The purpose of this report is to provide an overview of the models and overall system design of the Flask application. The application is a Grocery Store Webapp that allows users to browse and purchase grocery products. The system utilizes Flask, SQLite database, and follows the Model-View-Controller (MVC) architectural pattern.

2. Models

The application uses two main models: User and Product.

User Model:

The User model represents a registered user in the system.

It consists of the following attributes:

id: An integer representing the user's unique identifier.

Username: A string representing the username of the user.

Password: A string representing the user's password.

Product Model:

The Product model represents a grocery product available in the system.

It consists of the following attributes:

id: An integer representing the product's unique identifier.

Name: A string representing the name of the product.

ManufactureDate: A string representing the manufacturing date of the product.

ExpiryDate: A string representing the expiry date of the product.

Categoryid: An integer representing the category of the product.

rate: An integer representing the price of the product.

image: A string representing the image URL of the product.

cart: An integer representing the cart status of the product.

quantity: An integer representing the available quantity of the product.

unit: A string representing the unit of measurement for the product.

3. System Design

The overall system design follows the Model-View-Controller (MVC) architectural pattern.

Controller: The Flask application acts as the controller, handling the routing and logic for processing user requests.

It consists of various routes that are responsible for rendering views, processing form submissions, and interacting with the models and database.

Views: The views are responsible for displaying the user interface to the users.

HTML templates, rendered using Jinja2 templating engine, are used to generate dynamic content and present it to the users.

Views are designed to be user-friendly, responsive, and visually appealing.

Models: The models represent the entities in the system and define the structure of the data.

The User and Product models store user information and product details, respectively.

SQLite database is used to persist the data, and the models interact with the database using SQL queries.

4. Database Design

The application uses an SQLite database to store user and product information. The database consists of the following tables:

user Table:

Columns: id (integer), Username (text), Password (text)

admin Table:

Columns: id (integer), Username (text), Password (text)

category Table:

Columns: id (integer), Name (text), image (text)

product Table:

Columns: id (integer), Name (text), ManufactureDate (text), ExpiryDate (text), Categoryid (integer), rate (integer), image (text), cart (integer), quantity (integer), unit (text)

5. Conclusion

In conclusion, the Flask application follows the MVC architectural pattern and utilizes models to represent users and grocery products. The system design incorporates Flask, SQLite database, and HTML templates to create a functional and user-friendly Grocery Store Webapp.

Project Walkthrough Video :
https://drive.google.com/file/d/1Y3OUOaqMOwvIsZ4nvE6YHnXoSiho2Wzu/view?usp=drive_link