Content based GROCERY STORE

A project submitted in partial fulfilment of the requirement award of degree of

Diploma

In

COMPUTER ENGINEERING

Submitted by

R.ravi teja	23471-cm-085
P.abhi ram	23471-cm-077
P.venkat teja	23471-cm-119
T.rohith	23471-cm-104
SK.abdul haseeb	23471-cm-091
G.rohith kumar	23471-cm-123
T.mokshagna	23471-cm-098
P.arya	23471-cm-071
p.krishna	23471-cm-073

Under the esteemed guidance of

DEPARTMENT OF COMPUTER ENGINEERING

Pace institute of technology &sciences

(Approvedby AICTE , NewDelhi,Accredited byNBAand NCCAwith Agrade)

(Permanently Affiliated to Jawaharlal Nehru technological university. Kakinada)

Vallur NH-16,Ongole,prakasam district,AP-523272

Department of computer and engineering

CERTIFICATE

This is to certify that the project entitled GROCERY STORE is a bonafide work of R.ravi teja (23471- cm-085),P.abhiram (23471-cm-077),P.venkat teja (23471-cm-119),T.rohith (23471-cm-104),Sk.abdul haseeb (23471-cm-091),G.rohith kumar (23471-cm-123),T.mokshagna (23471-cm-098),p.arya (23471-cm-071),P.krishna (23471-cm-073) In the partial fulfilment of the requirement award of degree of bachelor of technology in Department of Computer and Engineering for the academic year 2023-2026.This work done by supervision and guidance

Signature ofGuidance.

Signature of Head of department
B.Satish kumar(M.tech)

Signature of the External Examiner

ACKNOWLEDGMENT

Wethankto almighty for giving us the course and perseverance in counting the main project. The project itself is Acknowledgment for all those people who have give as their heart-felt co-operation in making this project a grand success

We extend our sincere thanks to Dr.M.Venugopalrao. (BE,MBA) chairman of our college for providing sufficient infrastructure and good environment in the college to complete our course

We are thankful to our secretary Dr.M.Sridhar (mtech,MBA)

for providing the necessary infrastructure and labs and also permitting to carry out this project

We are thankful to our principal Mr. G. Koti reddy (M.tech,PhD) for providing the necessary infrastructure and labs and also permitting to carry out this project

We extreme jubilance and deepest gratitude, we would like to thank our head of the CME department Mr. B.Satish Kumar (M.tech)for his constant encouragement

We are greatly indebted to project guide (madam name) (B.tech) Lecturer, of computer engineering, for providing valuable guidance at every stage of this project work, we are profoundly grateful towards the unmatched services

rendered by him. Our special thanks to all the facility of computer engineering and peers for their valuable advises at every stage of this work

GROCERY STORE

R.ravi teja

P.abhiram

P.venkat teja

T.rohith

SK.abdul haseeb

G.rohith kumar

T.mokshagna

P.arya

P.krishna

GROCERY STORE

Introduction

Welcome to GROCERY STORE:

The Grocery Store web application is a comprehensive e-commerce platform built using Python Flask framework with SQLite database. This system provides a complete solution for online grocery shopping with separate interfaces for customers and administrators. The application follows the Model-View-Controller (MVC) architectural pattern and implements modern web development practices.

The application serves two primary user roles: customers who can browse products, add items to cart, and place orders; and administrators who manage products, view orders, and monitor store operations. The system features a responsive design that works across different devices and screen sizes.

Key Features:

- User registration and authentication
- Product catalog with search functionality
- Shopping cart management
- Order processing and tracking
- Admin dashboard for store management
- Image upload for products
- Real-time cart updates
- Order status tracking

SYSTEM ARCHITECTURE:

The Grocery Store application follows a three-tier architecture with clear separation between presentation, business logic, and data layers.

TECHNOLOGY STACK:

BACKEND:

- PYTHON 3.X
- FLASK WEB FRAMEWORK
- SQLITE DATABASE
- WERKZEUG FOR SECURITY AND FILE HANDLING

FRONTEND:

- HTML5 WITH JINJA2 TEMPLATING
- CSS3 WITH CUSTOM STYLING
- VANILLA JAVASCRIPT FOR DYNAMIC INTERACTIONS
- BOOTSTRAP FOR RESPONSIVE LAYOUT

Database Design and Data Management:

The database schema is carefully designed to support all aspects of grocery store operations while maintaining data integrity and performance. The schema consists of five main tables that work together to represent the core entities and relationships within the system. Each table serves a specific purpose and contains fields that capture essential information about the entity it represents.

The users table stores customer information including authentication credentials, contact details, and address information. This table forms the foundation of user management and enables personalized shopping experiences. The admins table maintains separate administrator accounts with enhanced security measures, ensuring that administrative functions remain protected from regular user access.

The products table serves as the product catalog, containing detailed information about each grocery item including name, price, description, category, and inventory levels. This table supports rich product presentations and inventory management capabilities. The orders table tracks customer purchases comprehensively, capturing not only the products ordered but also delivery information, payment methods, and order status throughout the fulfillment process.

The cart table manages temporary shopping data, allowing users to accumulate items before committing to a purchase. This table maintains the state of each user's shopping session and enables features like cart persistence across browser sessions. The relationships between these tables are carefully designed to maintain referential integrity while supporting efficient querying for various application functions.

FRONTEND IMPLEMENTATION AND USER EXPERIENCE:

The frontend of the Grocery Store application is built with a focus on creating an intuitive and engaging shopping experience. The interface uses HTML5 with Jinja2 templating for server-side rendering, CSS3 for styling and layout, and vanilla JavaScript for interactive elements. The design employs a consistent color scheme and visual hierarchy that guides users through the shopping process naturally.

The application uses a base template system that provides consistent layout elements across all pages, including navigation, footer, and common structural elements. This approach ensures visual consistency while reducing code duplication. The navigation system adapts based on user authentication state, showing relevant options for logged-in users, guests, and administrators.

Product presentation follows established e-commerce patterns with clear product images, prominent pricing information, and intuitive action buttons. The product grid layout responds to different screen sizes, ensuring optimal viewing experience regardless of device. Product cards include essential information at a glance while providing pathways to detailed product views and quick add-to-cart functionality.

The shopping cart interface provides a clear summary of selected items with options to modify quantities or remove items entirely. The cart calculates totals automatically, including subtotals, delivery charges, and grand totals, giving customers complete transparency about their purchase costs. The checkout process is streamlined to collect necessary information without creating unnecessary friction in the purchasing workflow.

HOW THE GROCERY STORE APPLICATION WORKS:

1. USER REGISTRATION AND AUTHENTICATION PROCESS:

The application begins with user registration and authentication. When a new customer visits the website, they can create an account by providing basic information such as username, password, address, and phone number. The system securely hashes the password using SHA-256 algorithm before storing it in the database, ensuring that plain text passwords are never saved. Once registered, users can log in using their credentials, and the system validates them against the stored hashed passwords. The authentication process checks both the regular users table and the administrators table to determine the appropriate level of access. Successful login creates a session that maintains the user's authenticated state throughout their browsing experience, allowing them to access personalized features like shopping cart and order history.

2. PRODUCT BROWSING AND SEARCH FUNCTIONALITY:

After logging in, customers enter the main product browsing interface where they can view all available grocery items. The products are displayed in an organized grid layout with each product showing an image, name, price, and an "Add to Cart" button. The system fetches product information from the database, including current stock levels, and only displays items that are available for purchase. Customers can use the search functionality to find specific products by entering keywords that match product names, descriptions, or categories. The search feature works by querying the database for partial matches across multiple fields, returning relevant results in the same intuitive grid layout. This approach helps customers quickly locate items they need without navigating through multiple categories.

3. SHOPPING CART MANAGEMENT SYSTEM:

When customers find products they want to purchase, they can add them to their shopping cart. The cart system works by storing product selections in the database linked to the user's session, allowing items to persist across browser sessions. Each time a customer adds a product the cart, the system checks current inventory levels to ensure the item is available. If theproduct is already in the cart, the system increments the quantity rather than creating duplicate entries. The cart interface displays all selected items with thumbnails, prices, quantities, and calculated line totals. Customers can adjust quantities using increase/decrease buttons or remove items entirely. The system automatically recalculates the subtotal, applies delivery charges, and displays the grand total in real-time as changes are made to the cart contents.

4. CHECKOUT AND ORDER PROCESSING WORKFLOW:

WHEN CUSTOMERS ARE READY TO COMPLETE THEIR PURCHASE, THEY PROCEED TO THE CHECKOUT PROCESS. THE SYSTEM FIRST VALIDATES THAT ALL ITEMS IN THE CART ARE STILL AVAILABLE AND IN SUFFICIENT QUANTITY. THE CHECKOUT FORM PRE-FILLS WITH THE CUSTOMER'S STORED ADDRESS AND PHONE INFORMATION, WHICH THEY CAN MODIFY FOR THIS SPECIFIC ORDER IF NEEDED. CUSTOMERS SELECT THEIR PREFERRED PAYMENT METHOD FROM OPTIONS LIKE CASH ON DELIVERY, CREDIT CARD, OR DIGITAL PAYMENTS. UPON SUBMITTING THE ORDER, THE SYSTEM PERFORMS SEVERAL CRITICAL OPERATIONS: IT RESERVES THE INVENTORY BY REDUCING STOCK LEVELS, CREATES ORDER RECORDS IN THE DATABASE, CALCULATES DELIVERY TIME ESTIMATES, AND CLEARS THE SHOPPING CART. THE ENTIRE PROCESS OCCURS WITHIN A DATABASE TRANSACTION TO ENSURE DATA CONSISTENCY—IF ANY STEP FAILS, ALL CHANGES ARE ROLLED BACK TO PREVENT PARTIAL ORDER CREATION OR INCORRECT INVENTORY COUNTS.

5. Order Fulfillment and Status Tracking:

After order placement, the system generates an order confirmation with details including order number, items purchased, delivery address, and estimated delivery time. Customers can view their order history through a dedicated interface that shows all past and current orders with their status. The order status progresses through various stages like processing, preparing for delivery, out for delivery, and delivered. Administrators can update these statuses through the admin dashboard as the order moves through the fulfillment pipeline. The system tracks each product within an order separately, allowing for partial fulfillment if some items become unavailable after order placement. Customers receive notifications at key status changes, keeping them informed about their order progress throughout the delivery timeline.

6. Administrative Operations and Store Management

Administrators access a separate dashboard that provides comprehensive store management capabilities. The admin interface displays key business metrics including total products, active orders, customer counts, and revenue indicators. Administrators can add new products to the catalog by filling out a form that includes product name, price, description, category, initial stock quantity, and product images. The image upload system validates file types and sizes before storing them in the designated upload directory. Product management features allow administrators to edit existing product information, update prices, modify descriptions, and adjust inventory levels. The system maintains a complete audit trail of product changes, though this history is not directly visible in the current interface.

7. Inventory Management and Stock Control

The application implements real-time inventory management that automatically updates stock levels when orders are placed. The system prevents customers from adding out-of-stock items to their cart and displays appropriate warnings when inventory is insufficient. When administrators add new products, they set initial stock quantities, and the system deducts from these quantities as orders are processed. The inventory checking occurs at multiple points: when adding to cart, during cart quantity adjustments, and at the final checkout stage. This multi-layered approach ensures that customers never order products that are unavailable. Administrators can view current stock levels through the product management interface and receive visual indicators when items are running low, though automated restocking alerts are not implemented in the current version.

8. USER SESSION MANAGEMENT AND SECURITY:

The application maintains user sessions to provide a continuous shopping experience without requiring repeated authentication. When users log in, the system creates a session identifier that is stored securely in an encrypted cookie on the user's browser. This session contains essential information like user ID, username, and role (customer or administrator). The system checks session validity on each page request to ensure users are properly authenticated for protected resources. Sessions automatically expire after a period of inactivity, requiring users to log in again for security. All sensitive operations, particularly those in the admin dashboard, include additional authorization checks to verify the user has appropriate privileges. The system also includes protection against common web vulnerabilities like SQL injection and cross-site request forgery.

9. PAYMENT PROCESSING AND ORDER COMPLETION

Log In to CookinBot:

Welcome back!Pleaseenter your credentials to access your account.

Email Address or Username:

Enter your registered email or username.

Password:

Enter your password.

(Forgot your password? [Click here](#) to reset.)

Remember Me:

[ ] Keep me logged in on this device.

Log In Button:

Click "Log In" to access your account.

New to CookinBot:

Don't have an account? [Sign up here](#) to join our community.

Need Help:

Having trouble logging in? Contact our [Support Team](#) for assistance.

9. PAYMENT PROCESSING AND ORDER COMPLETION

While the current implementation focuses on cash-on-delivery payments, the system is structured to support multiple payment methods. When customers select a payment option during checkout, the system records this preference with the order. For cash transactions, the payment status is implicitly pending until delivery completion. The architecture includes placeholder structures for integrating electronic payment gateways, though these are not fully implemented in the current version. Upon successful order creation, the system generates a unique order identifier that serves as a reference for both customers and administrators. This identifier helps track the order through the entire fulfillment process and resolves any customer service inquiries related to specific purchases.

10. Database Operations and Data Integrity

All application data is stored in a SQLite database with carefully designed tables and relationships. The system performs database operations using parameterized queries to prevent SQL injection attacks and ensure data security. Critical operations, particularly order creation and inventory updates, are wrapped in database transactions to maintain data consistency. This means that either all related database changes succeed together or fail together, preventing scenarios where inventory is deducted but no order is created. The database maintains relationships between users, products, carts, and orders through foreign key constraints, ensuring that references between tables remain valid. Regular database maintenance operations, though not explicitly implemented in the interface, would typically include backup procedures and integrity checks in a production environment.

11. Error Handling and User Feedback

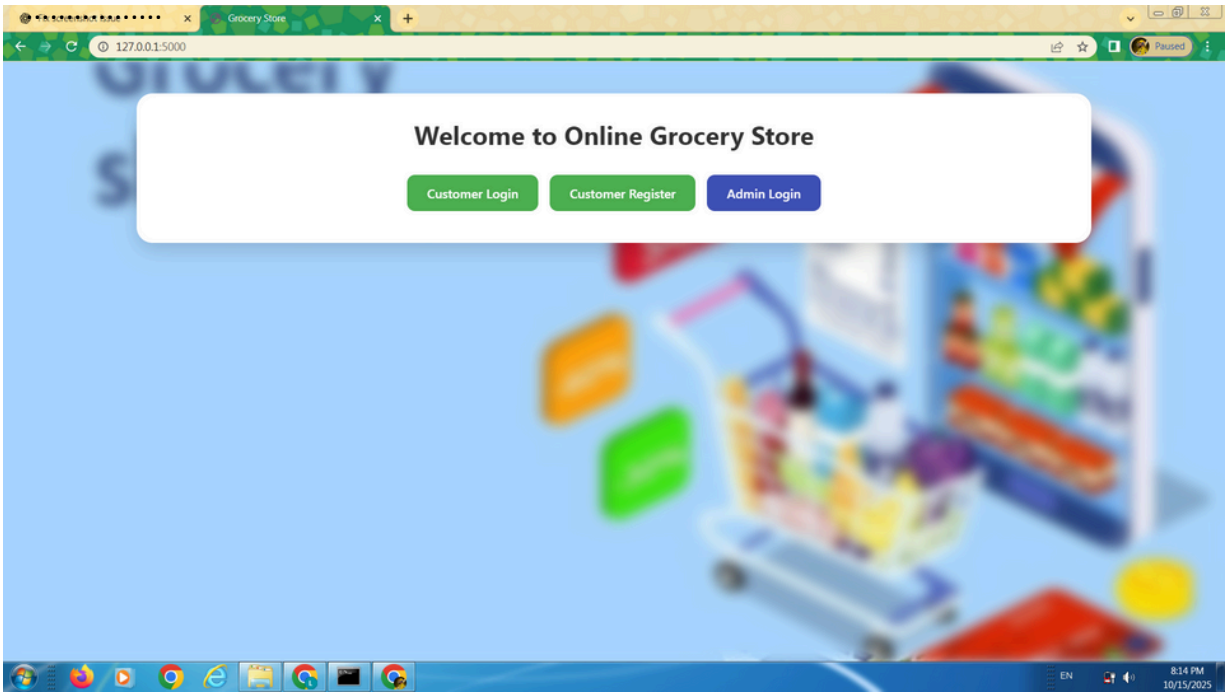
The application implements comprehensive error handling throughout all user interactions. When operations encounter problems, the system captures the error conditions and presents user-friendly messages without exposing technical details. Form validations check for required fields, data formats, and business rules before processing submissions. The flash message system provides immediate feedback to users about the success or failure of their actions, using different colors and styles to distinguish between information, success, warning, and error messages. Network connectivity issues are handled gracefully, with appropriate retry mechanisms for database operations. The system also includes fallback behaviors for missing product images and other optional content, ensuring the interface remains functional even when some resources are unavailable.

12. Responsive Design and Cross-Device Compatibility

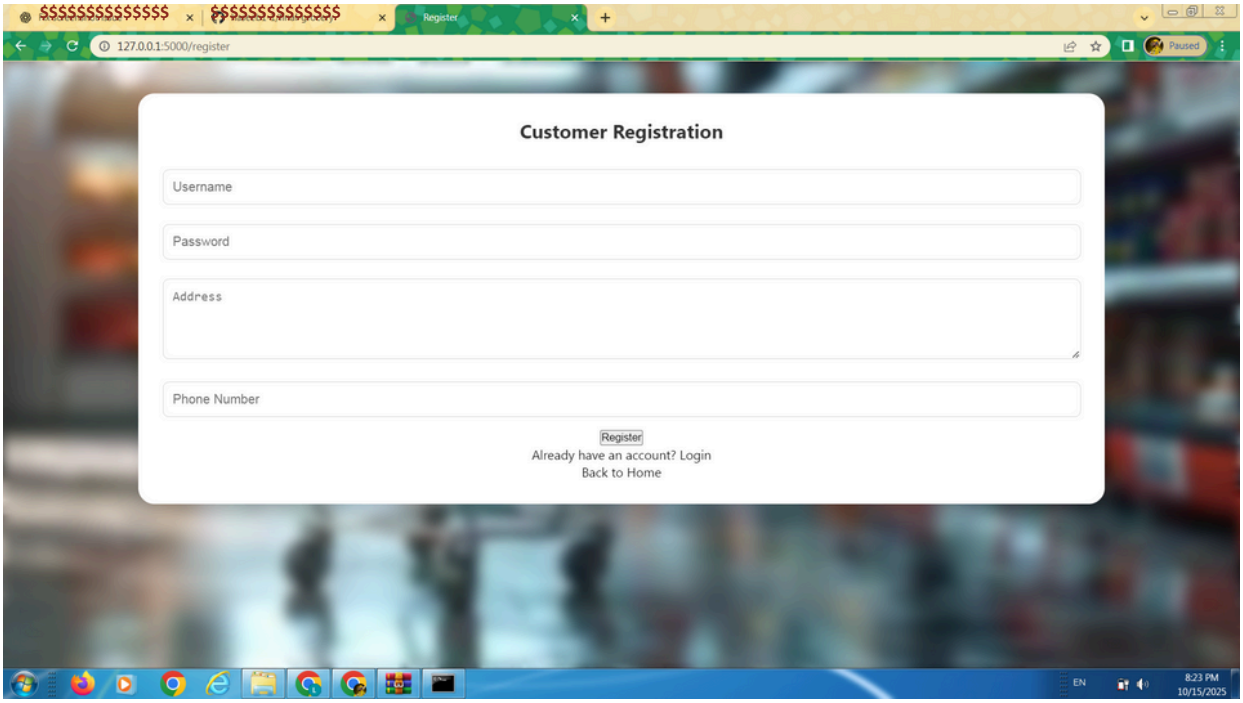
The user interface is designed to work seamlessly across different devices and screen sizes. The responsive design uses CSS media queries to adapt layouts, font sizes, and interactive elements based on the viewing device. On mobile phones, the product grid rearranges to a single column layout, navigation elements transform into mobile-friendly patterns, and touch targets are appropriately sized for finger interaction. The system maintains full functionality across devices, ensuring that customers can complete their grocery shopping regardless of whether they're using a desktop computer, tablet, or smartphone. This device flexibility is particularly important for grocery applications, as customers often shop using mobile devices while physically in stores or planning their shopping lists throughout the day.

The Grocery Store application works by integrating these various systems into a cohesive workflow that guides customers from product discovery through order fulfillment while providing administrators with the tools needed to manage store operations effectively. The architecture supports the typical e-commerce pattern while addressing grocery-specific requirements like inventory management, delivery scheduling, and fresh product considerations.

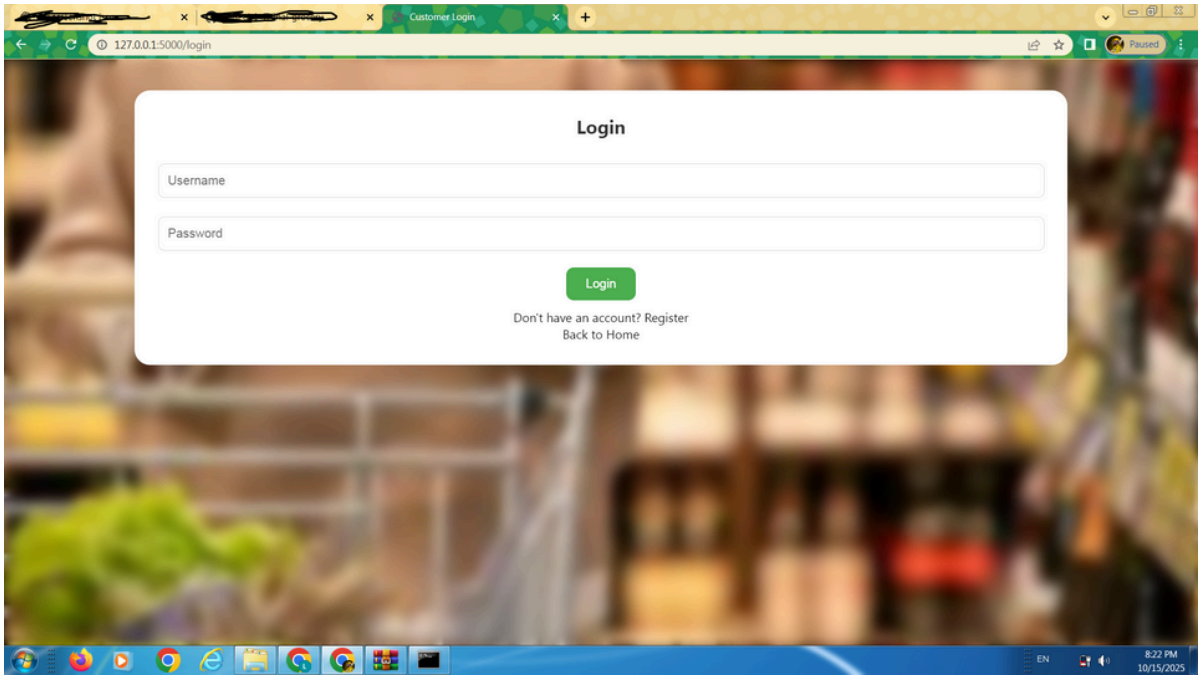
index page:



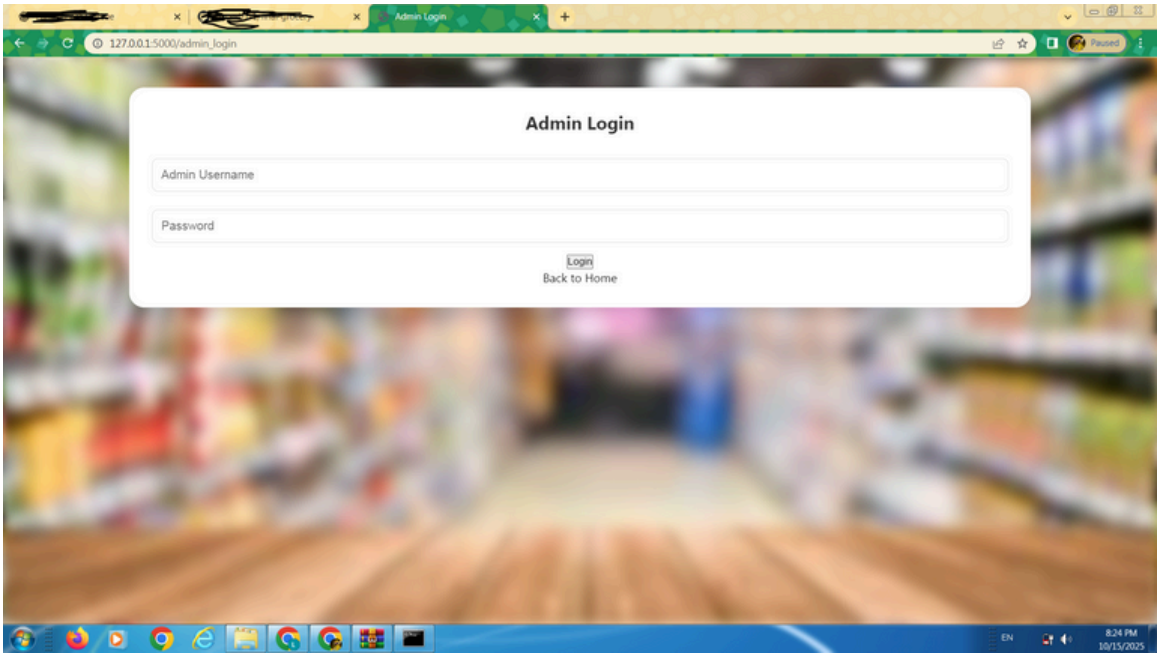
Register page:



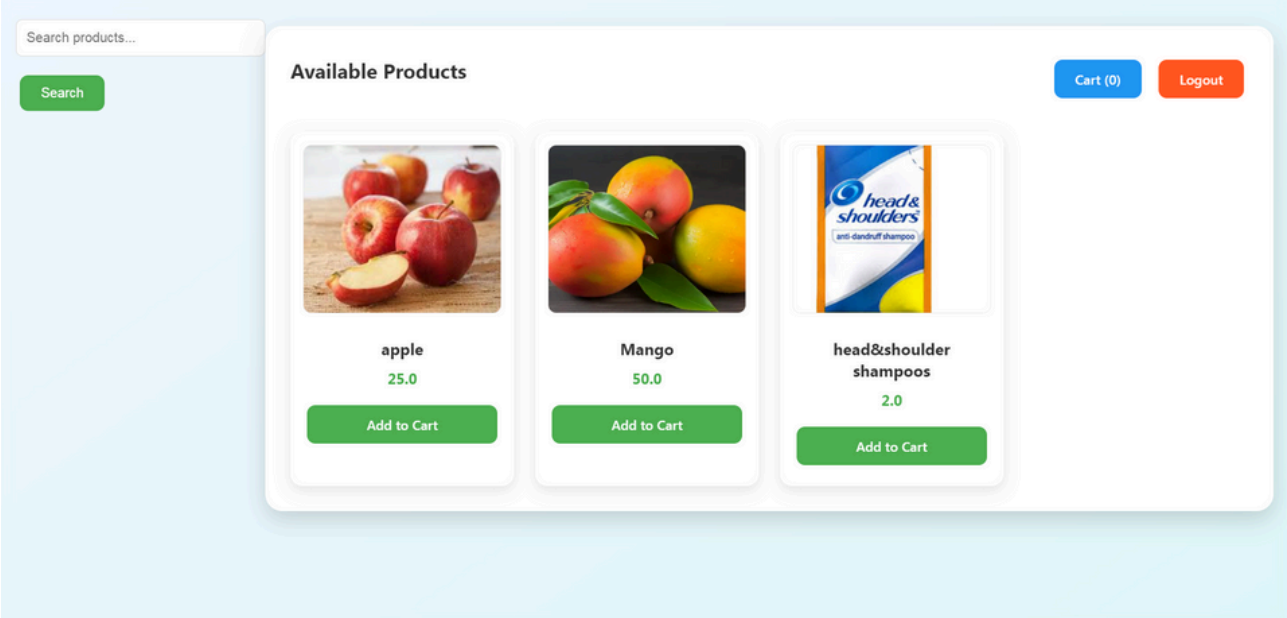
login page:



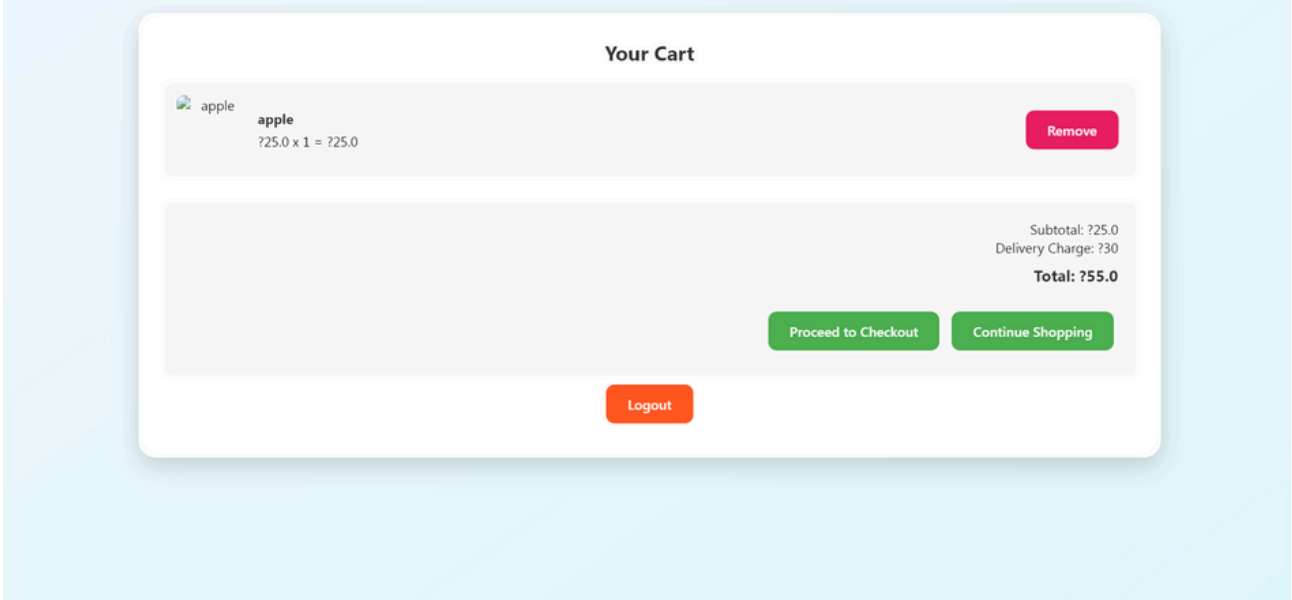
admin login:



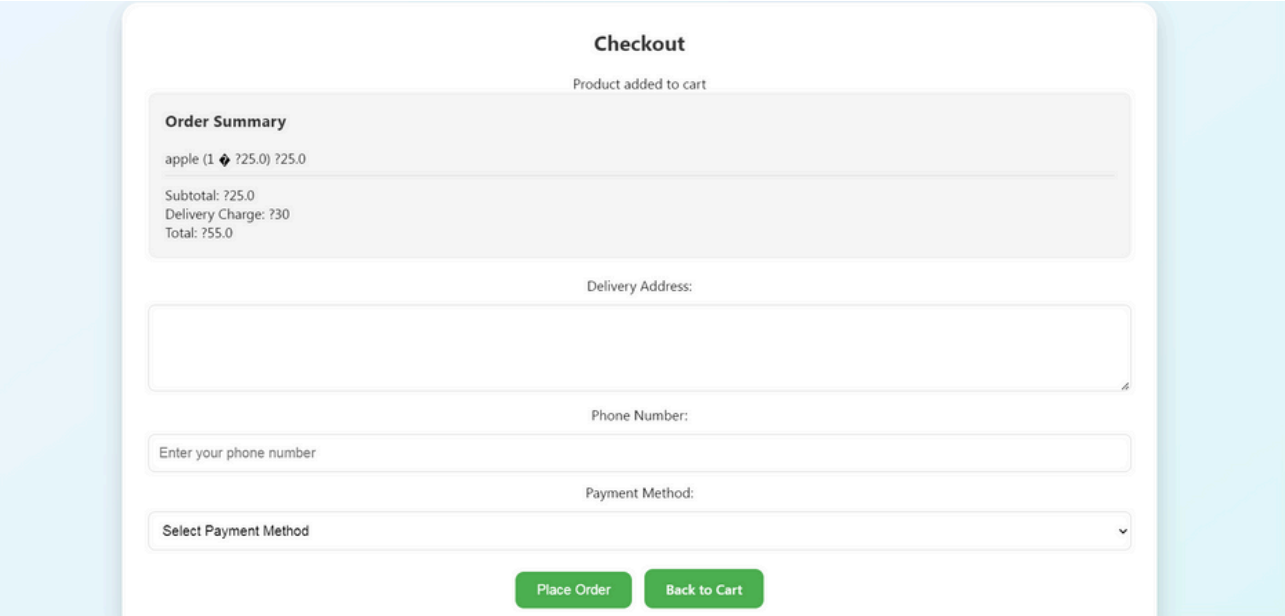
product page:



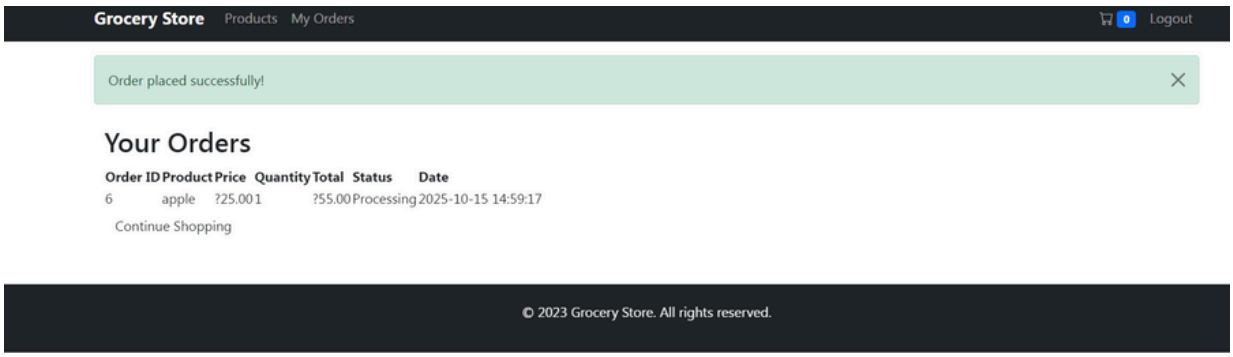
cart page:



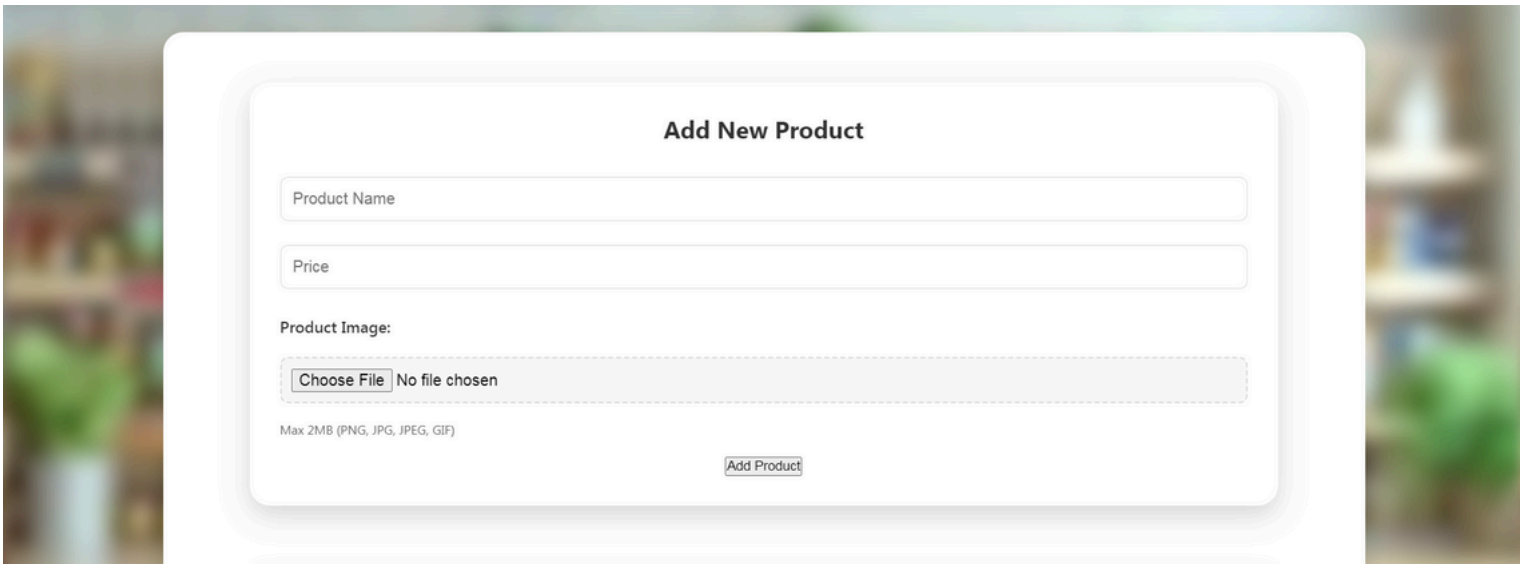
check out page :



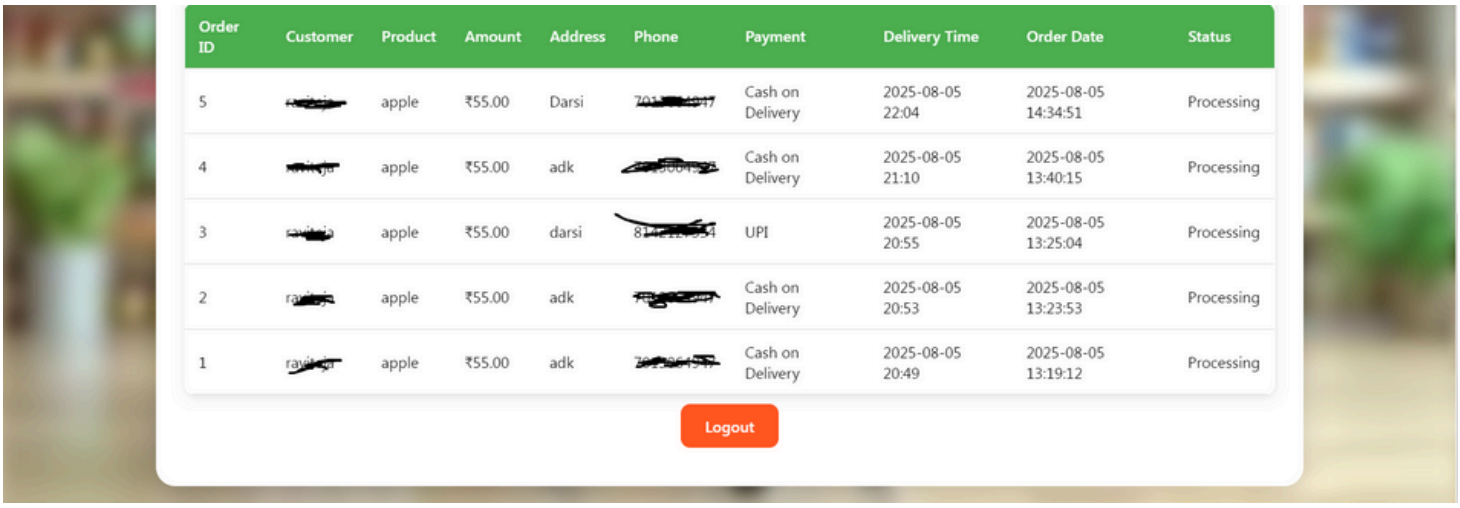
order placed page :



admin page(add new items):



admin page (orders list):



Order ID	Customer	Product	Amount	Address	Phone	Payment	Delivery Time	Order Date	Status
5	[redacted]	apple	₹55.00	Darsi	[redacted]	Cash on Delivery	2025-08-05 22:04	2025-08-05 14:34:51	Processing
4	[redacted]	apple	₹55.00	adk	[redacted]	Cash on Delivery	2025-08-05 21:10	2025-08-05 13:40:15	Processing
3	[redacted]	apple	₹55.00	darsi	[redacted]	UPI	2025-08-05 20:55	2025-08-05 13:25:04	Processing
2	[redacted]	apple	₹55.00	adk	[redacted]	Cash on Delivery	2025-08-05 20:53	2025-08-05 13:23:53	Processing
1	[redacted]	apple	₹55.00	adk	[redacted]	Cash on Delivery	2025-08-05 20:49	2025-08-05 13:19:12	Processing

Logout

SIMPLE STEPS TO RUN THE GROCERY STORE APPLICATION:

Step 1: Install Python

- Download and install Python from python.org
- Make sure to check "Add Python to PATH" during installation

Step 2: Install Flask

Open Command Prompt or Terminal and type:

pip install flask

Step 5: Run the Application

Open Command Prompt or Terminal

Navigate to your grocery_store folder:

cd path/to/your/grocery_store

step 4:Run the application:

python app.py

Step 5: Access the Application

- Open your web browser
- Go to: **http://localhost:5000**
- You should see the Grocery Store homepage

Step 6: Test the Application

1. As a Customer:
   - Click "Customer Register" to create an account
   - Login with your new account
   - Browse products and add items to cart
   - Checkout and place orders
2. As an Admin:
   - Click "Admin Login"
   - **Username: admin**
   - **Password: admin123**
   - Manage products and view orders

Step 7: Stop the Application

- Press Ctrl+C in the Command Prompt to stop the server

That's it! Your grocery store application is now running locally on your computer.

Conclusion:

Project Summary:

The Grocery Store web application represents a complete, functional e-commerce solution built using modern web development technologies. This project successfully demonstrates the implementation of a full-stack web application using Python Flask framework with SQLite database, providing a robust platform for online grocery shopping.

Key Achievements:

The application delivers a comprehensive shopping experience with separate interfaces for customers and administrators. Customers can seamlessly register accounts, browse products, manage shopping carts, and complete purchases, while administrators have full control over product management, inventory, and order processing. The system incorporates essential e-commerce features including user authentication, product catalog management, shopping cart functionality, order processing, and inventory tracking.

Technical Implementation:

From a technical perspective, the project showcases proper implementation of web application architecture following the Model-View-Controller pattern. The use of Flask framework provides a clean and maintainable codebase, while SQLite offers a lightweight yet powerful database solution. The frontend combines HTML templates with CSS styling and JavaScript functionality to create an intuitive user interface that works across different devices.

Security and Reliability:

The application incorporates important security measures including password hashing, SQL injection prevention, session management, and input validation. The transactional approach to database operations ensures data integrity, particularly during critical processes like order creation and inventory updates. Error handling and user feedback mechanisms provide a smooth user experience even when unexpected situations occur.

Learning Outcomes:

This project serves as an excellent example of full-stack web development, covering everything from database design and server-side programming to frontend implementation and user experience design. It demonstrates how different technologies can be integrated to create a cohesive, functional application that solves real-world problems.

Future Potential:

While the current implementation provides all core functionalities expected from an online grocery store, the architecture allows for future enhancements such as payment gateway integration, advanced analytics, mobile app development, and integration with third-party services. The modular design makes it relatively straightforward to extend functionality as requirements evolve.

Final Assessment:

The Grocery Store application stands as a testament to the power of combining simple, well-established technologies to create sophisticated solutions. It proves that with careful planning and proper implementation, even complex business processes can be effectively digitized and automated. This project not only functions as a practical grocery shopping platform but also serves as an educational resource for understanding web application development principles.

The successful execution of this project demonstrates that the fundamental concepts of e-commerce, user management, and inventory systems can be implemented efficiently using Python and web technologies, providing a solid foundation for more complex commercial applications in the future.