GroceryGo

Your Online Grocery Shopping Experience

CSE 428 Data Engineering

Team Members:
- Ahmed Hagag (120200101)
- Arwa Zakaria Khaled Alorbany (120200059)
- Mariam Ahmed Sheta (120200183)
- Youssef Ashraf Elharty (120200090)

Supervisors:
- Dr/Mohamed Abbasy
- Eng/Ahmed Hesham

Contents:

1. Introduction:

GroceryGo store is a website designed to offer a shopping experience for multiple users through the integration of databases with web development technologies (Django framework). The key features are user management, shopping cart functionality, grocery listing, advanced filtering options, search capabilities, and purchase analytics.

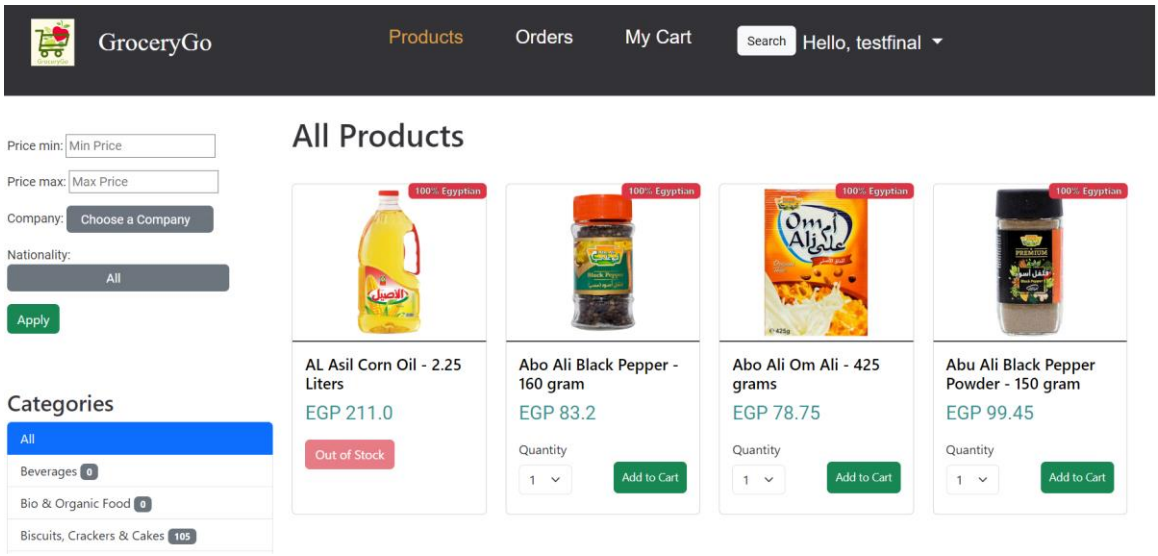

Figure 1 Website's home page

2. Implemented Features:

User Management and Shopping Cart:

User Authentication

We developed the website so that users register and log in with unique credentials, resulting in personalized accounts. Users can reset password via email sent from our store's management. We created a user's table in the database so that each user is associated with their own shopping cart, ensuring a personalized shopping experience. We created profiles for each user to manage additional data for users i.e. additional addresses.

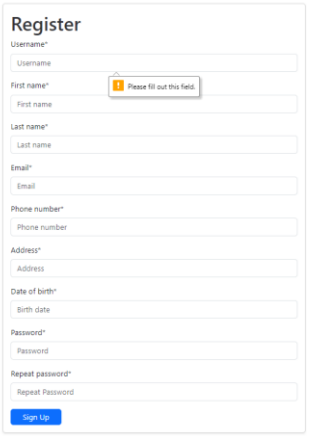

Figure 2 register form.

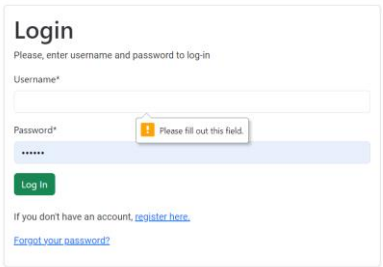

Figure 3 log in form

User account management:

Each user can update profile data. We created an additional addresses table with one-to-many relationship with the users table to manage more than one address for each user. Users can choose between all their addresses when placing an order, ensuring a better shopping experience.

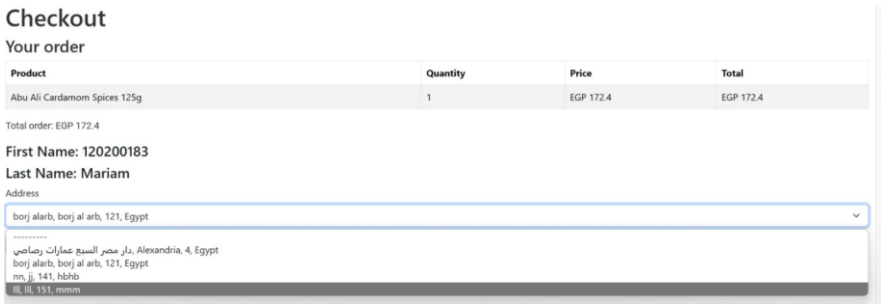

Figure 4 Select different addresses option.

Shopping Cart Database

We created a database table dedicated to storing user shopping carts. This table records the items added, their quantities, and the total cost for each individual user. Users can view cart details with all their purchases. Furthermore, they can update, add, delete from the shopping cart before proceeding to placing the order. Coupons are also applied.

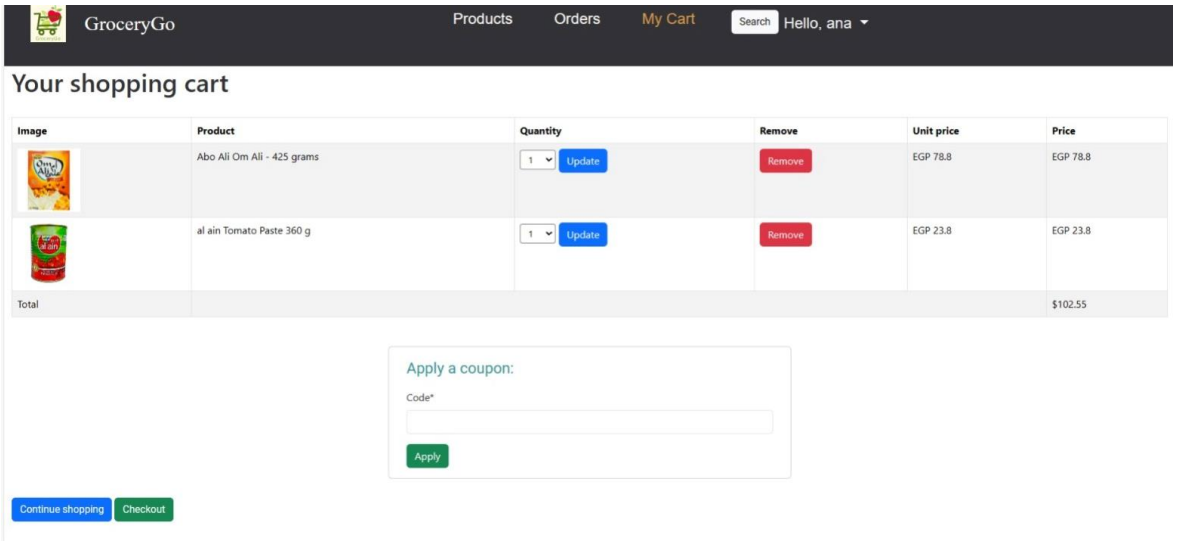

Figure 6 Personal cart page

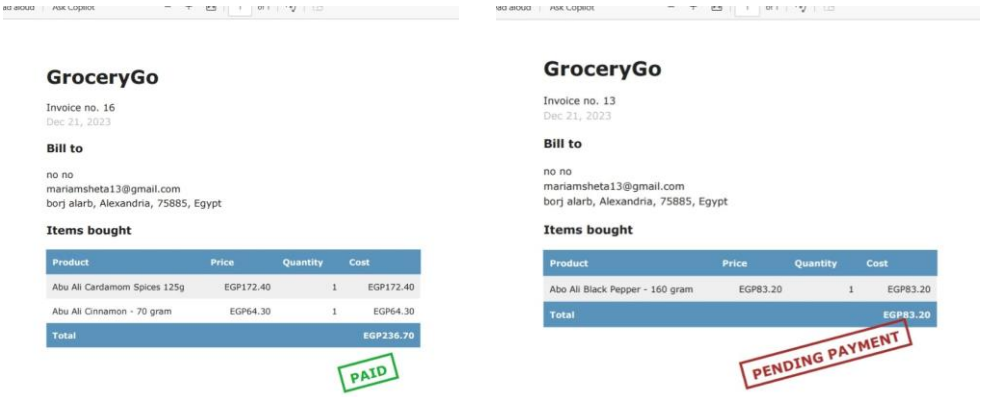

Figure 5 Payment pdf

Grocery Listing and Filtering:

Grocery Database and listing:

We created a database for the available products in our shop. Items were sectioned according to category, company, and the products provided by each company. Each product details include name, price, brand and its specialties, category, and the brand nationality. Users can navigate between categories provided in the website sidebar.

Filtering and search Options:

To streamline the shopping process, users can filter the grocery list based on various criteria, including price range, company name, and brand nationality. This ensures that users can tailor their shopping experience to their preferences. About 90% of the website's products are 100% Egyptian.

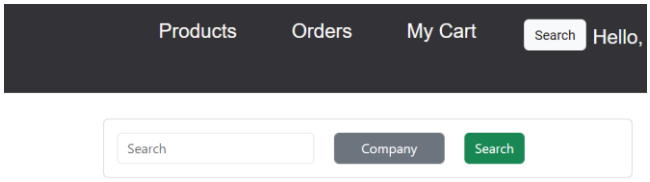

Figure 8 Search bar

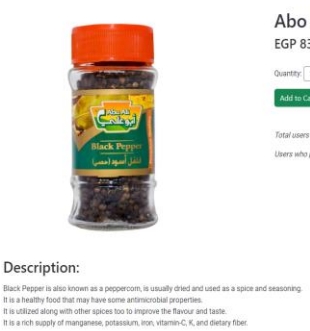

Figure 9 Product's details

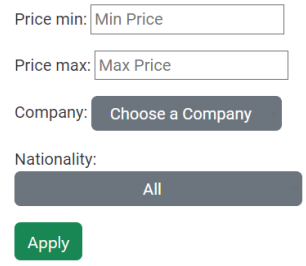

Figure 10 Search by price, company, and nationality.

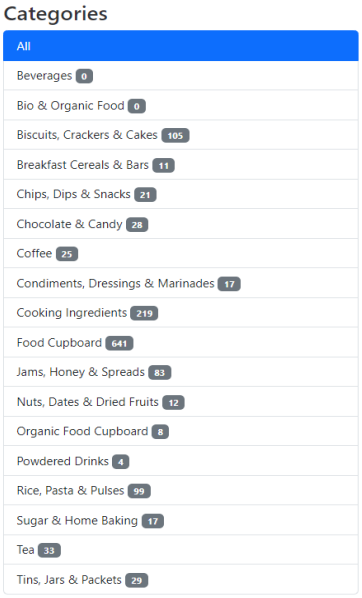

Figure 7 Categories available.

Purchase Analytics

Our website provides valuable insights into product popularity by displaying the product details and the number of users who have purchased a specific item. This information is presented both overall and within the last 24 hours, giving users a sense of trending products.

Figure (11) 100% Egyptian

3. The application’s endpoints:

Our website consists of 6 apps with the following endpoints:

Main app:

- Root URL:
  - The root URL immediately redirects users to the shop section of the website.
- Home:
  - The home endpoint leads to the home page of the site.
- Admin:
  - Provides access to Django's built-in admin interface for site management.
- Account:
  - This view includes various account-related functionalities.
- Orders:
  - This endpoint is where orders history is viewed.
- Shop:
  - The endpoint redirects to the shop for
- Cart:
  - This endpoint shows cart details.

Account app:

- Login Page:
  - This endpoint provides users access to the login page. Users can authenticate themselves to access their accounts.
- Logout:
  - The logout endpoint allows users to securely log out of their accounts.
- Password Reset:
  - This view leads to the password reset page.
- Password Reset Done:
  - After a user requests a password reset, they are redirected to this page, which confirms that the password reset email has been sent.
- **Register:**
  - This page is for new users to register for an account through this endpoint.
- **Account View:**
  - This is the default landing page for user's account. It typically displays the user's profile information, order history, or other personalized content.
- **Profile Update:**
  - This endpoint facilitates profile information, like name, email, or address**.**
- **Add Address:**
  - This endpoint allows users to add a new address to their profile.

Shop app:

- **Product List:**
  - This endpoint displays a general list of products available in the shop. This is typically the main shopping page where all products are displayed.

- **Product List by Category:**
  - This URL is used to display products filtered by specific categories.
- **Product Detail:**
  - This endpoint provides detailed information about a specific product, including descriptions, pricing, and availability.

Cart app:

- Cart Detail:
  - This endpoint displays the details of the user's shopping cart. It shows the items currently in the cart, their quantities, and prices.
- Add to Cart:
  - This endpoint is used for adding a product to the shopping cart. It handles the addition of a product to the user's cart, adjusting quantities as needed.
- Remove from Cart:
  - This endpoint allows users to remove a product from their shopping cart.

Orders app:

- Order Create:
  - This endpoint is used to initiate the creation of a new order.
- Order Created:
  - This endpoint is used to confirm and display the details of the created order. This endpoint provides users with reassurance that their order has been successfully placed and processed.
- Admin Order PDF:
  - This endpoint is designed for administrative purposes to generate a PDF document of a specific order. It is especially useful for creating printable invoices or receipts.

Coupons app:

- Apply Coupon:
  - This endpoint allows users to apply a coupon code to their order. This feature enhances the shopping experience by offering promotional discounts.
- Remove Coupon:
  - This URL is used for removing a previously applied coupon from an order.

4. Database Schema:

Schema Structure and Functionality

User Authentication and Authorization:

This is the Central to the application are tables like auth_permission, auth_group, auth_group_permissions, auth_user, auth_user_groups, and auth_user_user_permissions. These are standard Django tables that manage user permissions and roles, ensuring secure and flexible user access control.

User Profiles and Addresses:

The account_profile and account_address tables store user information and addresses. These tables are used for creating personalized shopping experiences, enabling users to manage their profiles and delivery addresses.

Product Management:

The grocery shop's product management is dependent on tables shop_category, shop_company, and shop_product. These tables collectively manage the grocery products' categorization, company or brand information, and product specifics.

Shopping Cart and Order Processing:

The core of our website functionality lies in cart tables. The cart_cart and cart_cartitem tables manage the customers' shopping carts, storing information about the products added to the cart. The orders_order and orders_orderitem tables manage checkout, handling the order details and the individual items within each order.

ERD Diagram:

The following design shows ERD diagram for the website database. Each table is created to represent distinct entities within the grocery shopping website, representing the relationships between users, products, and orders. The link for the diagram is link.

Relationships:

- Each user is associated with one profile (One-to-One relationship).
- Users can have multiple addresses, each associated with a unique user (One-to-Many relationship).
- Users can have multiple shopping carts, each associated with a unique user (One-to-Many relationship).
- Products belong to a specific category (Many-to-One relationship).
- Each product can be associated with multiple orders (Many-to-Many relationship).
- Each company has multiple products (one-to-many relationship).
- Each user can make one order or more (One-to-Many relationship)

5. Going live:

In the development of GroceryGo, we used Heroku, a cloud-based Platform as a Service (PaaS), renowned for its simplicity in deployment and infrastructure management. This choice allowed us to focus on building the application with Python, without the overhead of server and hardware concerns. Heroku's straightforward Git integration facilitated smooth code deployment, while its scalability features enabled us to efficiently handle varying traffic demands. Additionally, Heroku's rich ecosystem of add-ons provided essential services like database management and performance monitoring, significantly enhancing our development process. Ultimately, Heroku's blend of user-friendliness, scalability, and comprehensive tool support was instrumental in the successful launch and maintenance of the GroceryGo project.

6. Technologies Used:

Web Development:

- Frontend: HTML and CSS
- Backend: Django (Python-based web framework).

Database Integration:

- Utilization of a relational database (SQLite).
- Implementation of Django ORM for database interactions.

7. Project Tasks management:

Division of Tasks:

Account Application - Arwa Zakaria Khaled Alorbany

Responsibility: Arwa was tasked with developing the 'Account' application, a crucial component of GroceryGo.

Key Contributions: She focused on implementing user authentication, profile management, and password reset functionalities.

Cart Application - Ahmed Haggag

Responsibility: Ahmed took charge of the 'Cart' application, an integral part of the user shopping experience.

Key Contributions: His work involved creating and managing the shopping cart functionality. This included developing features for adding, updating, and deleting items from the cart, as well as integrating real-time updates to the cart's contents.

Shop Application - Youssef Ashraf Elharty

Responsibility: Youssef was responsible for the 'Shop' application, the core interface for product display and interaction.

Key Contributions: His primary focus was on implementing the product listing, search, and filter functionalities.

Orders Application - Mariam Ahmed Sheta

Responsibility: Mariam was assigned the 'Orders' application, a vital component handling the order processing and management.

Key Contributions: She developed features related to order placement, tracking, and history. Mariam’s commits indicate meticulous attention to ensuring accurate order handling and customer satisfaction.

Collaborative Approach:

The team's approach to dividing tasks was based on a combination of individual skills and project requirements. The GitHub commits serve as a testament to each member's dedication and the collaborative effort put into different aspects of the GroceryGo project. Regular updates, feature enhancements, and bug fixes were part of the ongoing development process, ensuring a well-rounded and efficient application.

Bonus Taks:

- Simulate the payment process. (Mairam)
- Showing some cool features. show random discounts and add additional addresses. (Arwa)
- Showing the user if it’s 100% Egyptian product through shareholders nationality of each brand (you can use API, web scraping or just local table for that) to encourage the national industry. (Ahmed and Youssef)

8. Links:

GroceryGo website link:

https://grocery-go-24-b1f88daa13b9.herokuapp.com/

GitHub repository for the project:

https://github.com/AuroraaSan/GroceryGo