الجامعة المصرية اليابانية للعلوم والتكنولوجيا

E-JUST

Egypt - Japan University of Science and Technology

エジプト日本科学技術大学

CSE 428

Data Engineering

Grocery Shop Project

Tech Trolly

Submitted by

Mariam Ayman – 120200094

Asem Mohamed – 120200089

Mostafa kotb – 120200043

Ashrakat saeed – 120200091

Abdelrahman said - 120200075

TO

Dr. Mohamed Abassy

Eng. Ahmed Hesham

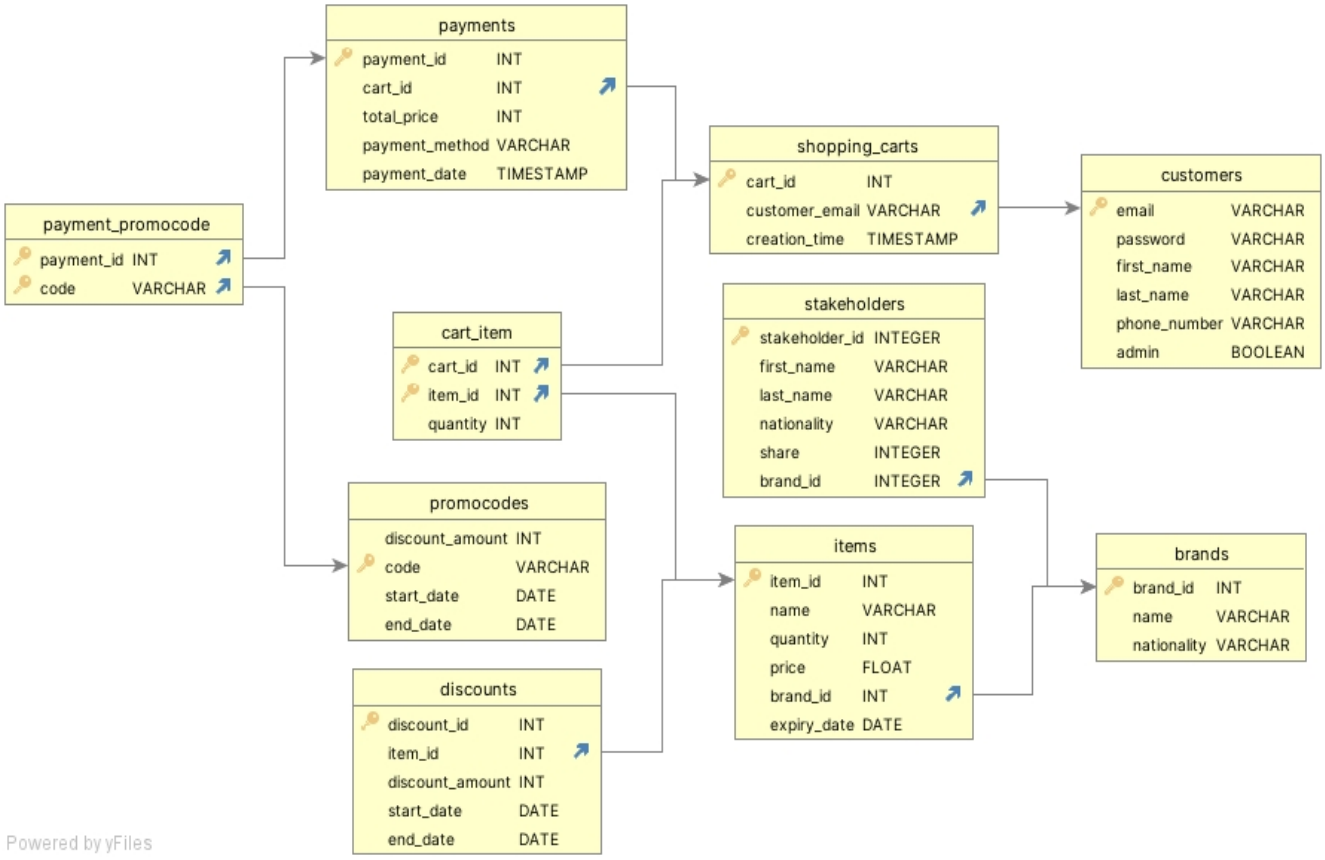
Table of Contents

Project Overview

This project, named 'Tech Trolly,' is a desktop application designed to serve as a virtual grocery shop, as an Amazon clone but tailored for desktop use. Developed using Python, Tkinter, and SQLite, Tech Trolly offers users an intuitive and efficient shopping experience. This documentation will provide a detailed breakdown of each component, offering insights into the development process and the unique features of the application.

Database Schema

The following figure shows the schema with its 10 tables including the primary and foreign keys in each of them as well as the relations between them:



Database Generation

The following libraries to generate a local data to fill in the local tables of our database: Faker, Numpy, Hashlib, and Random. In the below sections, each generation file is discussed in details:

- **items.py**

  This file consists of the following 3 functions:

  generate_expiration_date:

  - Purpose: Generate a random expiration date within the range of 1 to 5 years from the current date.
  - Return: A string representation of the generated expiration date.

  generate_supermarket_item_name:

  - Purpose: Select a random category and then choose an item name associated with that category, creating a realistic item name for a supermarket.
  - Return: A string representing the generated item name.

  generate_realistic_items_insert_statements:

  - Purpose: Generate SQLite insert statements for items with realistic data taking the number of records as an input and producing a list of SQLite insert statements accordingly.
  - Parameters: num_records (int): The number of insert statements to be generated.
  - Return: A list of SQLite insert statements for items.

- **customers_shopping_carts.py**

  This file consists of the following 4 functions:

  generate_password_hash:

  - Purpose: Hashes the provided password using SHA-256 algorithm.
  - Parameters: password (str): The password to be hashed.
  - Return: A string representing the hashed password.

  generate_phone_number:

  - Purpose: Generates Egyptian phone numbers starting with +2010, +2011, +2012, +2015.
  - Return: A randomly generated phone number.

  generate_customer_insert_statements:

- Purpose: Creates insert statements for the 'customers' table with realistic data.
- Parameters: num_records (int): The number of insert statements to be generated.
- Return: A list of insert statements for customers.

generate_shopping_carts_insert_statements:

- Purpose: Creates insert statements for the 'shopping_carts' table with realistic data.
- Parameters: num_records (int): The number of insert statements to be generated.
- Return: A list of insert statements for shopping carts.

- **payment_promocode_promocodes.py**

  This file consists of the following 4 functions:

  generate_payment_promocode_insert_statements:

  - Purpose: Creates insert statements for the 'payment_promocode' table, associating payments with promo codes.
  - Parameters: num_records (int): The number of insert statements to be generated.
  - Return: A list of SQL insert statements for payment-promocode associations.

  generate_promo_code:

  - Purpose: Generates a random alphanumeric code of length 6 for promo codes.
  - Return: A string representing the generated promocode.

  generate_discount_amount:

  - Purpose: Generates a random discount amount between 10 and 100.
  - Return: An integer representing the generated discount amount.

  promocodes_insert_statements:

  - Purpose: Creates insert statements for the 'promocodes' table with realistic data.
  - Parameters: num_records (int): The number of insert statements to be generated.
  - Return: A list of insert statements for promocodes.

- **stakeholders.py**

  This file consists of the following 2 functions:

  generate_nationality:

  - Purpose: Generates a random nationality for diversity in brand origins.

- Return: A string representing the generated nationality.

generate_customer_insert_statements:

- Purpose: Creates insert statements for the 'stakeholders' table, simulating stakeholders associated with various brands.
- Return: A list of insert statements for stakeholders.
- Function Details:
  - ✓ For each of the 20 iterations, a unique random brand ID is generated.
  - ✓ Stakeholders are generated with random first names, last names, nationalities, and ownership shares in the associated brand.
  - ✓ The 'share' attribute denotes the ownership percentage, and the sum of shares for each brand is constrained to 100%.

- **collected_insert_queries.py**

  This file includes all the code from the previous files with the calling functions to them to generate the data.

Main Features

1. User Management:

   There are 2 types of users: Admin and Customer.

   - User
     - ✓ Implement user authentication and authorization.
     - ✓ Each user should have unique credentials.
     - ✓ Users should have their own shopping carts.
   - Admin
     - ✓ Each admin can add new items to the grocery store as well as adding other admins.

2. Shopping Cart:
   - Users can add items to their shopping carts.
   - Users can remove items from their shopping carts.
   - Users can view the contents of their shopping carts.

3. Grocery List:
   - Display a list of available groceries in the shop.
   - Implement filtering options based on:
     - ✓ Price range
     - ✓ Brand name
     - ✓ Brand nationality

- ✓ Category
- ✓ On Discount

4. Search Functionality:
   - Users can type in the name of the grocery, category, or brand to search for a specific item.

5. Product Statistics:
   - Show the number of users who have purchased a specific product in total.
   - Show the number of users who have purchased a product in the last 24 hours.

Bonus Features

1. Simulated Payment Process:
   - Implemented a simulation of the payment process with options: credit card, debit card, and cash on delivery.
   - If the credit card option is selected, the application prompts the user to input Visa details and performs verification using Luhn Algorithm.
2. Cool Features Showcase:
   - Introduced random discounts ranging from 5% to 25% on products set to expire in one week.
   - Implemented a filter option allowing users to only view products with active discounts.
3. National Product Identification:
   - Utilized shareholders' nationalities associated with each brand to identify products as 100% Egyptian.
   - Implemented a visual indicator by coloring 100% Egyptian products in green for easy identification.
   - Introduced a filter option labeled "Nationality" to display products specifically from Egyptian brands.

Running the Project

1. Install Dependencies:
   - Run the following command to install required dependencies:

   "pip install Tkinter SQLite3 Random Numpy Hashlib Faker Pillow Screeninfo Datetime"

2. Download and Run:
   - Download the project from GitHub.
   - Navigate to the project directory and run: "python application.py"

3. Enjoy the Application:
   - The "application.py" file manages all app pages.
   - Start enjoying the grocery shop application!

Teamwork Division

- Database Schema
  In the first meeting, we all designed the database schema.

- Database Generation
  The generation of the local database was distributed among the members as following:
  - Mariam: items.py and stakeholders.py
  - Assem: brands&discounts.py
  - Mostafa: customer&shopping_carts.py
  - Ashrakat: cart_item&payments.py
  - Abdelrahman: payment_promocode&promocodes.py

- Application Pages
  The pages of the application were distributed among the members as following:
  - Mariam: payments.py
  - Assem: cart.py
  - Mostafa: home.py
  - Ashrakat & Abdelrahman: login.py & signup.py & admin.py

- Bonus Features
  These features were implemented in another meeting by all of the team members.

- UI
  The pages of the application were distributed among the members as following:
  - Mariam: signup.py & layout of home.py
  - Ashrakat: payment.py & layout of home.py
  - Mostafa: login & UI of the card item in home.py
  - Abdelrahman: cart.py & images and logos of the home.py
  - Assem: admin.py & UI of 100% Egyptian displayed products in home.py