KLE Society's

KLE Technological University



**Open Ended (OE) Assessment Report**

**On**

**SUPER MARKET BILL GENERATING SYSTEM**

**Object Oriented Programming (22ECSC204)  
Object Oriented Programming Lab (22ECSP203)**

Submitted by

| **Name** | **Roll no** | **SRN** |
| --- | --- | --- |
| Tarun S M | 108 | 01FE21BCI008 |
| Suresh H B | 115 | 01FE21BCI015 |
| **Team Number: 7** | | |

Faculty In-charge:

Ms.Vijayalaxmi

SCHOOL OF COMPUTER SCIENCE & ENGINEERING

Academic year 2022-23

| **1.** | **Introduction** | |
| --- | --- | --- |
|  | 1.1 | Overview of the problem statement |
|  | 1.2 | Features of Application |
| **2.** | **Design** | |
|  | 2.1 | Class Diagrams |
| **3.** | **Description** | |
|  | 3.1 | Classes |
|  | 3.2 | Main Function |
| **4.** | **Implementation** | |
|  | 4.1 | Results |

|  |  | |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  | |
|  |  |  |
|  |  | |
|  |  |  |

1. **Introduction**

The Supermarket Management System is a console-based application that simulates the operations of a supermarket. It allows customers to add products to their shopping carts, generate bills, and also provides administrative functionalities for managing the product inventory**.**

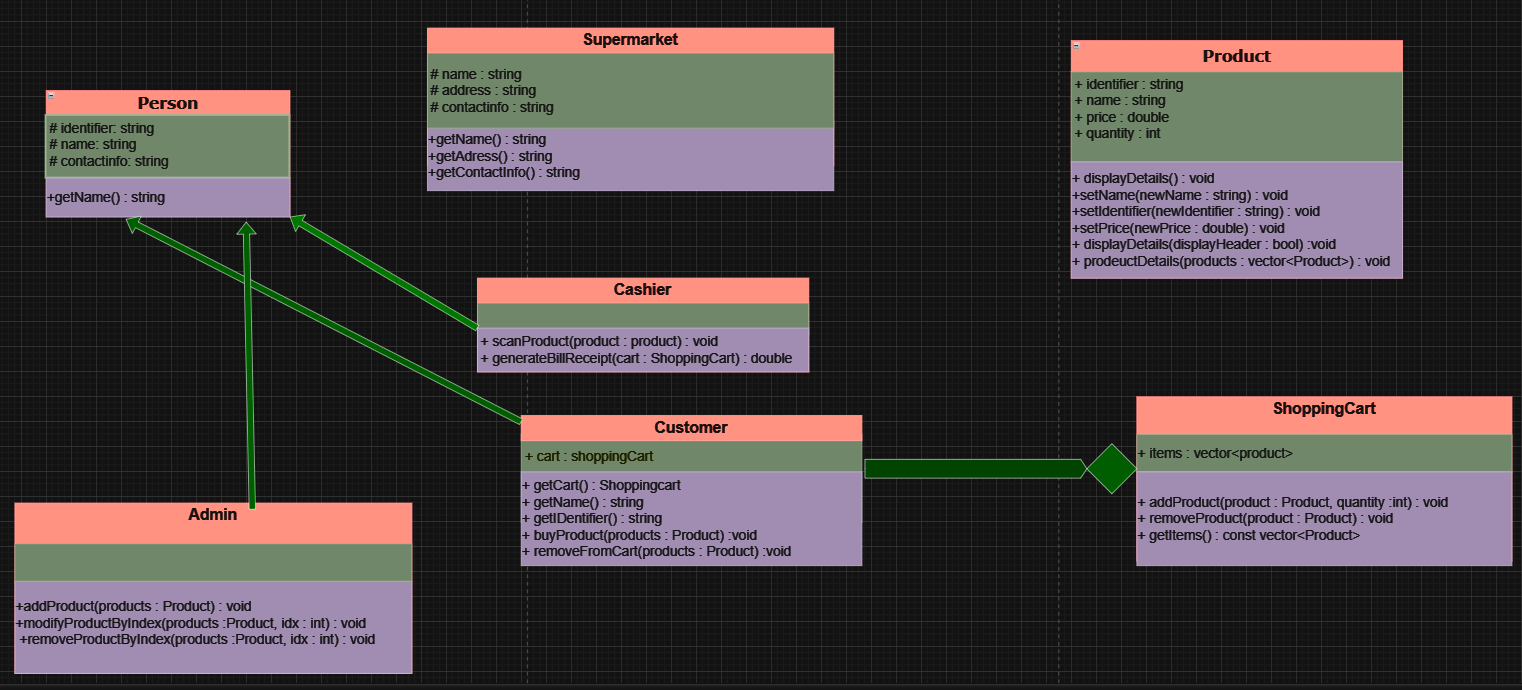
**1.1 Overview of the problem statement**

The task is to develop a dual interface Super-Market Bill generating System. The system should allow customers to purchase products, remove Items From their Cart and to give their feedback. The system should also support admin authentication and access create Product and delete product and modify the price, stock ,name of the products.

**1.2 Features of Application**

* The Supermarket Management System offers the following features:
* **Customer Functions:**
* Add products to the shopping cart.
* Remove products from the shopping cart.
* Generate bill receipt for the purchased items.
* View cart.
* Update or reduce the quantity in cart.
* **Administrator Functions:**
* Add new products to the inventory.
* Modify product details (name, price, quantity) in the inventory.
* Delete products from the inventory.
* View the current stock of products.
* View feedback

1. Design
   1. Class diagram :-



* 1. Design Pattern :

The design pattern used in the above code is the Model-View-Controller (MVC) design pattern. MVC is an architectural pattern commonly used in software development to separate an application into three interconnected components: the Model, the View, and the Controller. Each component has a specific role, which helps in organizing and managing the codebase effectively.

* Model :-The Product class represents the model in the MVC pattern. It encapsulates the data related to products, including their identifier, name, price, and quantity. The model is responsible for managing the data and providing methods to access and modify it (e.g., getter and setter methods).
* View :- The admin class and the Cashier class are responsible for the view part of the MVC pattern. They provide methods for displaying information to the user. For example, the admin class provides methods to display product details and stocks, while the Cashier class provides methods to generate a bill receipt.
* Controller :- The main function and the menu function act as the controller in the MVC pattern. The controller handles user input, interacts with the model and view, and controls the flow of the application. It takes user input, makes decisions based on that input, and triggers appropriate actions on the model and view.

1. Description

3.1 Classes :-

1. Person :- The Person class represents a general person and has three data members: identifier (to store a unique identifier for the person), name (to store the person's name), and contactInfo (to store the contact information of the person). The constructor initializes these data members, and the getName() function returns the person's name.
2. Supermarket :- The Supermarket class represents a supermarket. It has three data members: name (to store the supermarket's name), address (to store the address of the supermarket), and contactInfo (to store the contact information of the supermarket). The constructor initializes these data members, and the displayDetails() function is used to display the supermarket's details.
3. Product :- The Product class represents a product available in the supermarket. It has four data members: identifier (to store a unique identifier for the product), name (to store the product's name), price (to store the price of the product), and quantity (to store the quantity available in the supermarket). The constructor initializes these data members, and the class provides various member functions to display, set, and get the product details.
4. ShoppingCart :- The ShoppingCart class represents a shopping cart for a customer. It has one data member: items (a vector of Product objects representing the products added to the cart). The class provides member functions to add products to the cart, display the cart items, remove products from the cart, and update the quantity of items in the cart.
5. Customer :- The Customer class is a subclass of Person and represents a customer. It has one additional data member cart (an object of the ShoppingCart class). The class provides member functions to buy/add products to the cart, remove products from the cart, view the cart, and update the quantity of items in the cart.
6. Cashier :- The Cashier class is a subclass of Person and represents a cashier in the supermarket. It has only static member functions scanProduct() and generateBillReceipt(). These functions take a customer's cart as input and perform the actions of scanning the products and generating the bill receipt.
7. Admin :- The admin class contains static member functions to perform administrative tasks in the supermarket, such as adding products, modifying product details, deleting products, viewing product stocks, and viewing customer feedbacks
8. errorHandler Class :- This class is a simple error handling class. It has two data members: id (to store the error status code) and msg (to store the error message). The constructor initializes these data members, and the printError() function is used to display the error information.

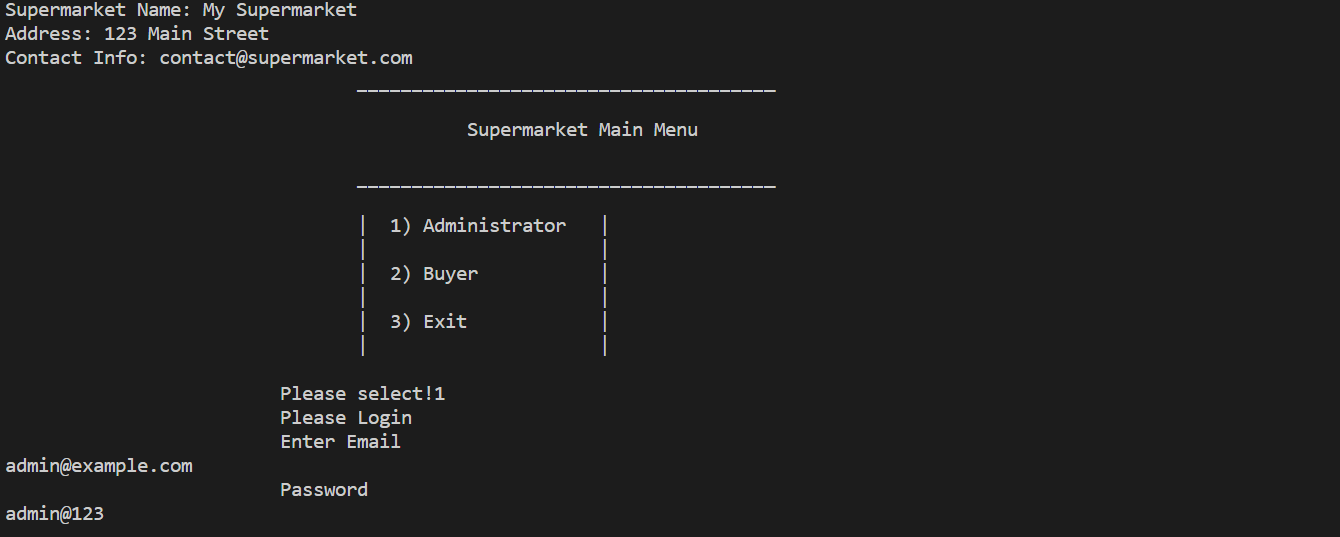
3.2 Main Function :-

The main function sets up the initial state of the supermarket, creates a list of products, and then enters an infinite loop where customers can interact with the supermarket through the menu function. The loop continues until the program is manually terminated.

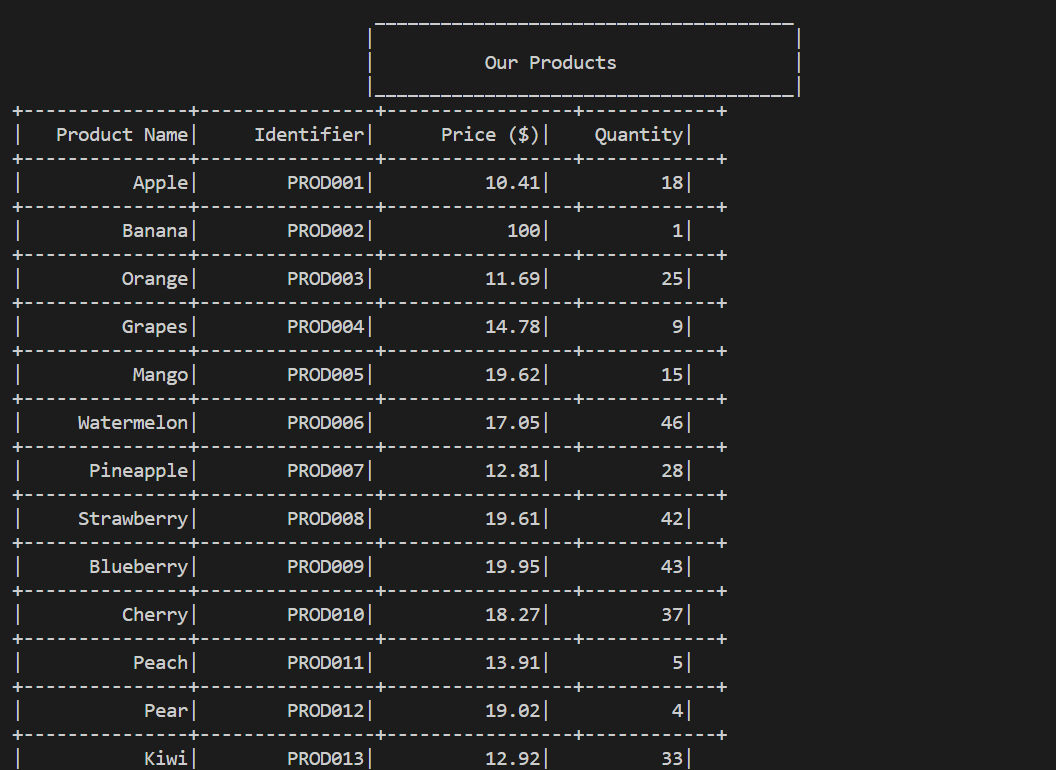
1. Vector of Feedbacks: A vector of pairs (vector<pair<int, string>> feedbacks) is declared to store feedbacks given by customers. Each pair consists of an integer (int) representing the customer index and a string (string) representing the feedback provided by that customer.
2. Create Supermarket: An instance of the Supermarket class is created with the name "My Supermarket," address "123 Main Street," and contact email [contact@supermarket.com.](mailto:contact@supermarket.com.).
3. Display Supermarket Details: The displayDetails method of the Supermarket class is called to show the details of the supermarket on the console.
4. Create Vector of Products: A vector of Product objects, named products, is declared. However, the code provided does not show the actual code to populate this vector with product details. Instead, it indicates that the products vector should be created with appropriate Product objects representing different products available in the supermarket.
5. Infinite Loop for User Interaction: The code enters an infinite loop with while (1) to allow continuous interaction with the user. Inside the loop, the menu function is called repeatedly to handle user input and execute different functionalities of the supermarket system.
6. Menu Function Call: The menu function is called with the arguments products, supermarket, and feedbacks. This function handles the main user interface and allows users to perform actions such as buying products, viewing the cart, generating bill receipts, and more.
7. Exit Code: The return 0; statement is provided to indicate that the main function has completed successfully. In this code, it is unreachable due to the infinite loop, and the program will terminate only when the user chooses to exit the application from the menu function.
8. **Implementation**

**4.1 Results**

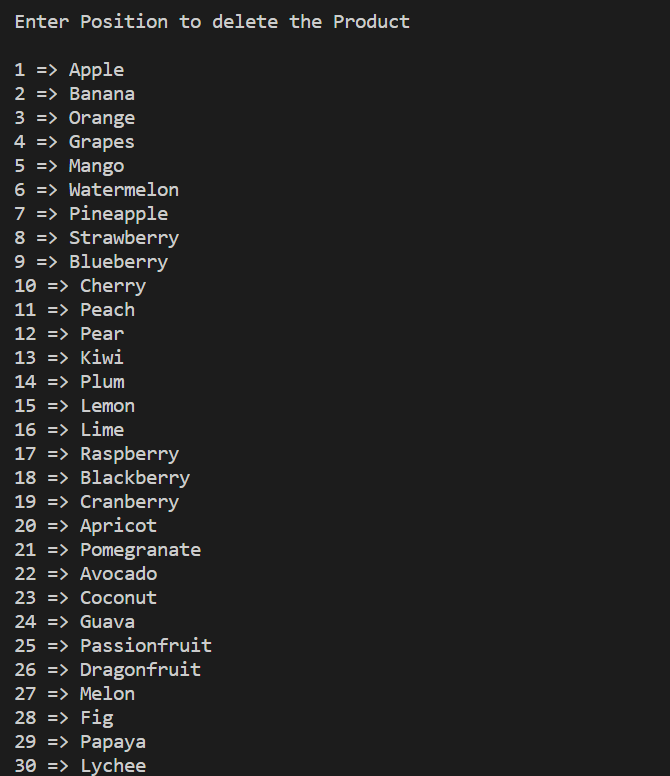
* **Initial interface**

****

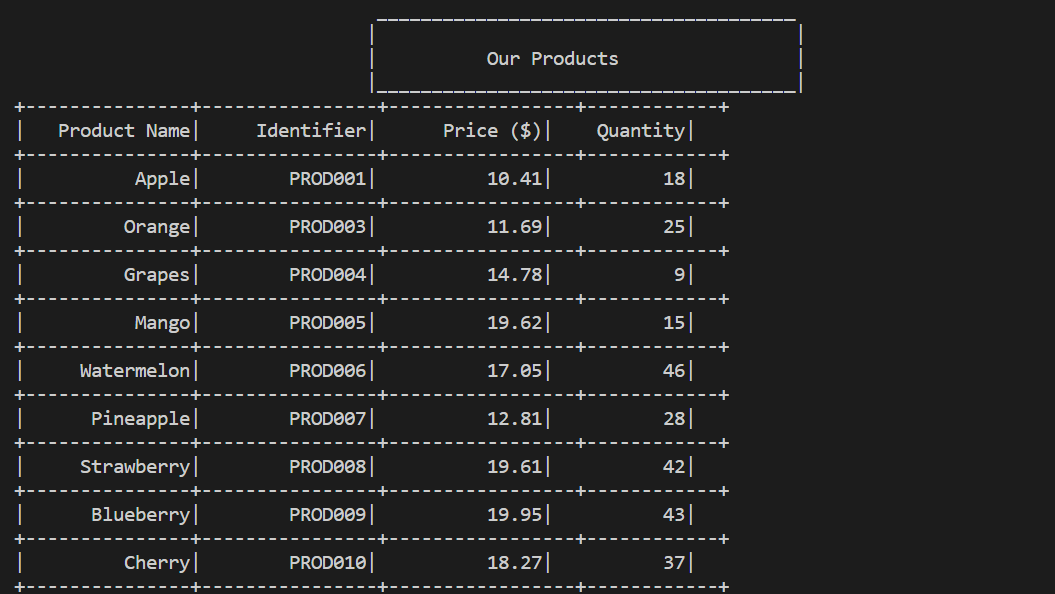
* Products viewed by admin

****

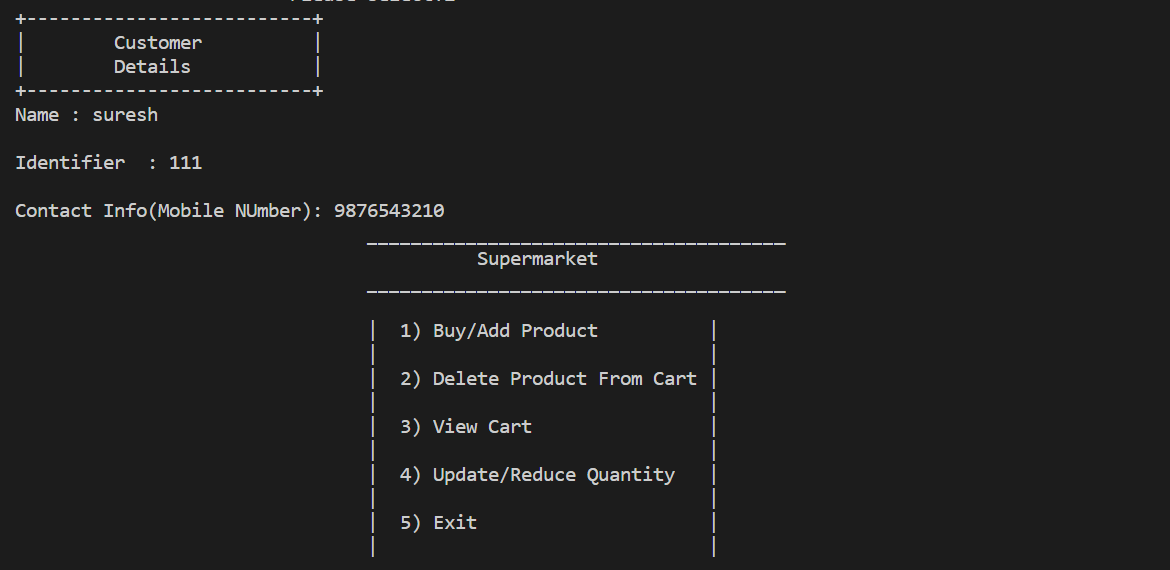
* **Deleting the product from stock**

****

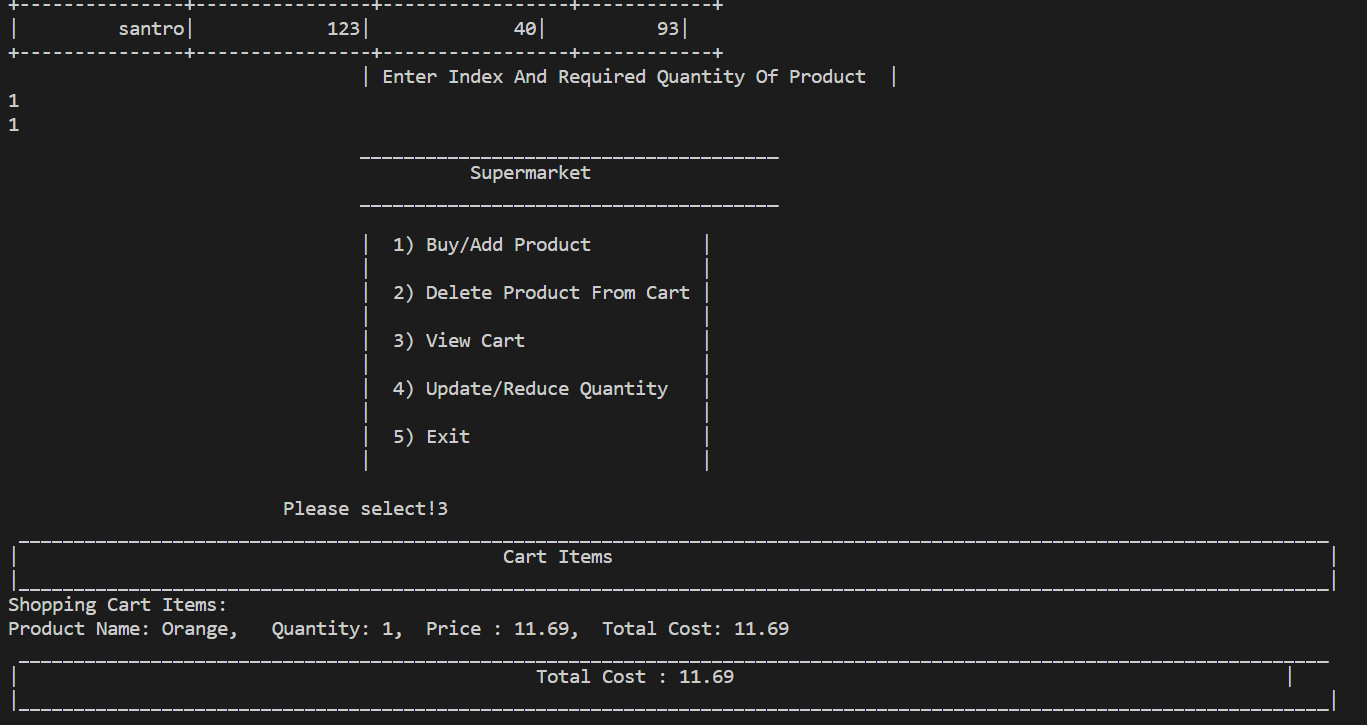
* **Out put after deleting the product(banana is deleted)**

****

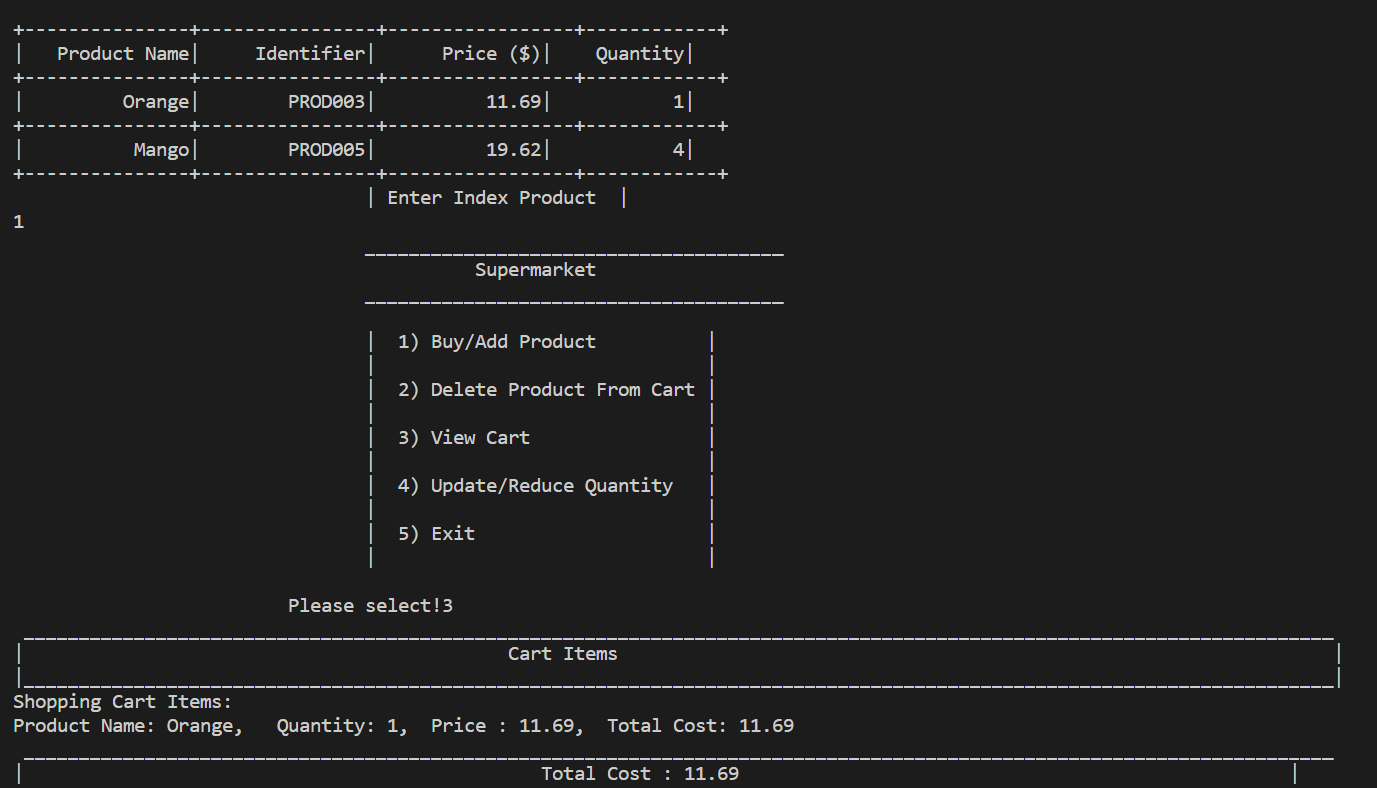
* **Login interface for user**

****

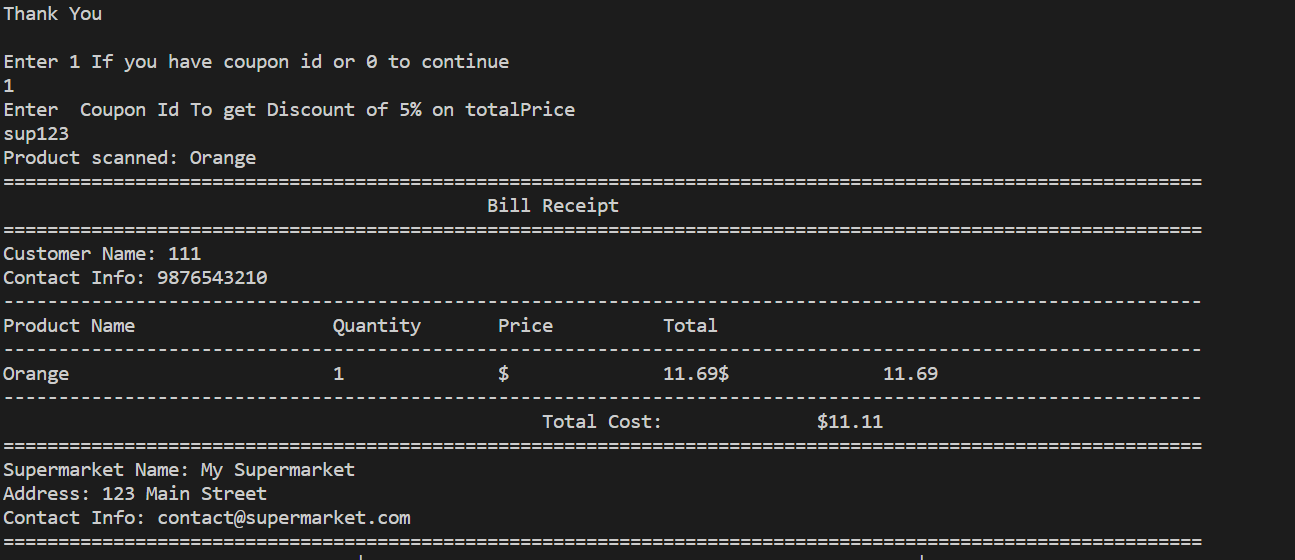
* **User buying the product**

****

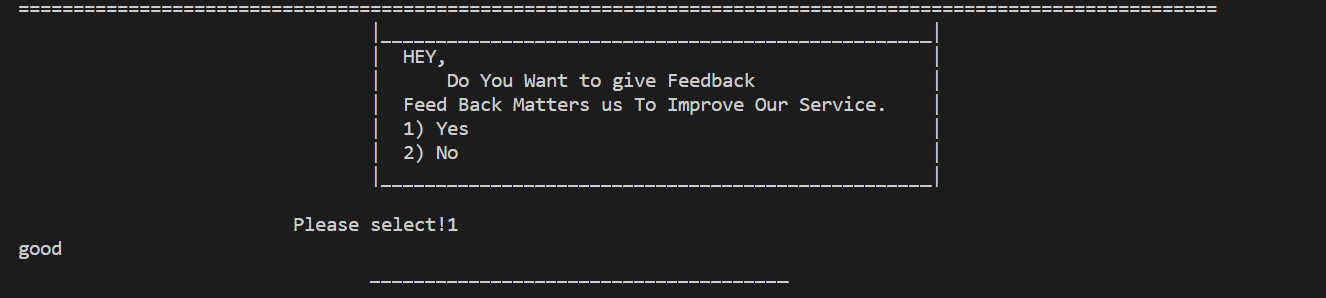
* **Deleting the product from cart**

****

* **Bill generating**

****

* **Giving feedback**

****