

GROCERY MANAGEMENT SYSTEM

SOFTWARE DEVELOPMENT-ASSESSMENT\_02



January 29, 2024

UNIVERSITY OF LAW

BLOOMSBURY CAMPUS LONDON

**Grocery Management System**

**Table of Contents**

1. Introduction

* Purpose and Significance
* Why Python as the programming language

1. Grocery Management System Main Menu
   1. Product Management Main Menu
   2. Order Management Main Menu
   3. Customer Management Main Menu
   4. Point of Sale (POS) System
   5. Inventory Control
2. Testing and Quality
3. Future Work
4. Conclusion

**Introduction**

In the retail industry, the Grocery Management System is a key tool for improving and streamlining grocery shop operations. Successful inventory management, order processing, and customer interactions are critical components of the grocery industry's dynamic and competitive landscape. These issues are addressed by the Grocery Management System, which provides a centralized platform for managing inventory, orders, clients, and other crucial elements of grocery shop operations.

* *Purpose and Significance*

The main goal of the Grocery Management System is to give managers and owners of grocery stores an advanced tool for efficiently running their businesses. This covers order fulfilment, inventory management, and customer service—all essential elements of the food retail ecosystem. Through the integration of critical features and the automation of repetitive processes, the system minimizes human error and effort while increasing total productivity.

The potential of the Grocery Management System to transform conventional grocery store models is what makes it significant. Python's fundamental characteristics of adaptability, readability, and ecosystem made it the ideal programming language for developing the Grocery Management System.

**Grocery Management System Main Menu**

1. **Product Management Main Menu**

A collection of choices for managing the products in the supermarket inventory is offered by the Product Management Menu. These choices consist of adding a new product, looking at the ones that are already there, changing product information, taking a product out of stock, and making a purchase.

*Function*

* *Add Product (`add\_product`):* The user can add a new product to the inventory using this feature. The product's name, quantity, and price are requested from the user, with the necessary validations applied for numeric inputs. The snapshot of the function implemented is shown in Fig. 1.

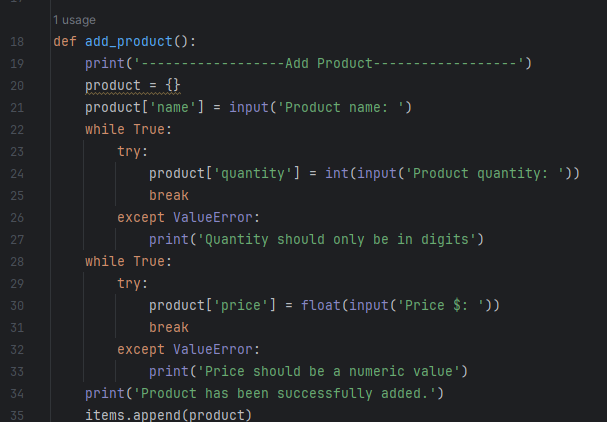
**

Figure 1

* *View Items (`view\_items`):* This feature shows every item that is in the inventory. It gives information on each product, including its name, price, and quantity. Python code of the function can be seen in Fig. 2.

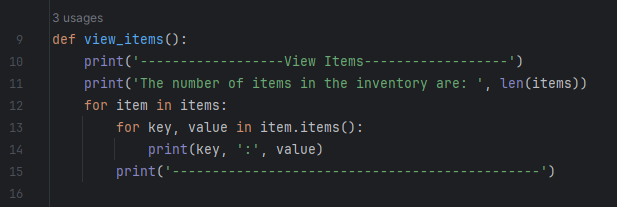
**

Figure 2

* *Update Product (`update\_product`):* The user can update the details of an existing product with this feature. When a product that needs to be modified is located, the user is prompted to input its name and can then change its name, quantity, and price. The screen of the update product code is shown below in Fig. 3.

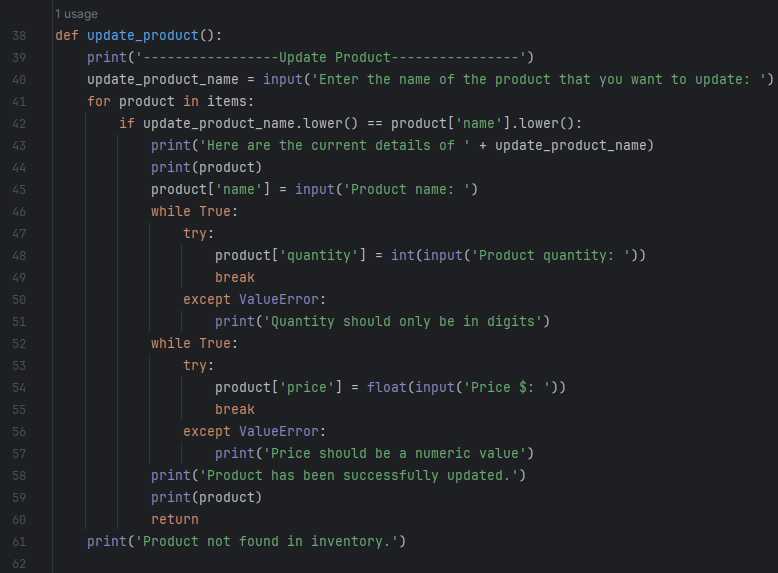
**

Figure 3

* *Remove Product (`remove\_product`):* A product can be taken out of the inventory using this function. If the product is located, the user is prompted to input its name, and it is then deleted from the list of items. The Code Snapshot is shown in Fig. 4.

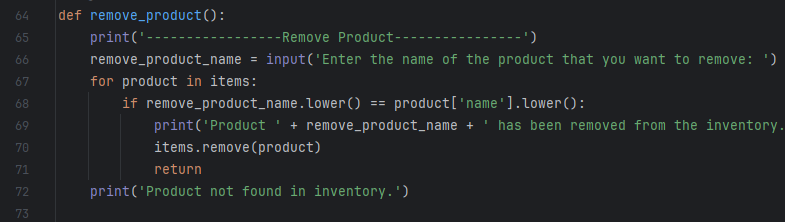
**

Figure 4

* *Purchase Product (`purchase\_product`):* This feature mimics making a product purchase. When a user enters the name of the product they wish to buy, a buy message is shown and the quantity is lowered if the product is available. Screen of the function is shown in Fig. 5.

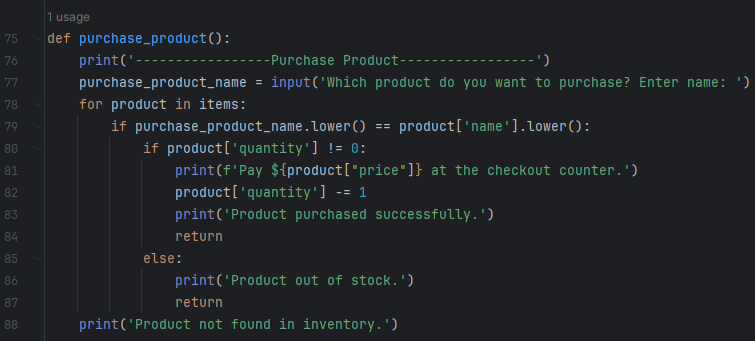
**

Figure 5

**Integration**

Together, these features enable a product's full lifespan within the inventory, from addition and viewing to updating, removing, and simulating a purchase.

1. **Order Management Main Menu**

Managing client orders is the main function of the Order Management Menu. Order details can be updated, new orders can be placed, and existing orders can be viewed.

*Functions*

* *Place Order (`place\_order`):* The new order placement process is started by this function. When adding products to the order, the user is required to input their names. After then, the order is included in the list of orders.
* *View Order (`view\_order`):* This feature shows all of the current orders' details. It offers an organised picture of every order, including with the products that are part of the order.
* *Update Order (`update\_order`):* A customer can update the specifics of an existing order with this function. The user adds or removes products from the order after selecting the order to be modified and providing its number.

**Integration**

When combined, these features make it easier to create, examine, and update customer orders. This is necessary for effectively tracking and completing client requests.

1. **Customer Management Main Menu**

Managing customer data and purchasing history is the main emphasis of customer management. The primary choices consist of

*Functions*

* *Add Customer (`place\_order`):* By entering their name and email address, new customers can be added with the help of this feature. It makes the system more capable of keeping an extensive customer database inbuilt data type as in the list.
* *View Customer (`view\_customer`):* provides a thorough view of every client, including with email addresses and names. This feature offers a concise and well-structured summary of the clientele.
* *Update Customer (`update\_customer`):* lets users edit client information, such as email and name. This guarantees that client data is accurate and current.
* *View Purchase History (`view\_purchase\_history`):* Displays the past purchases made by a certain consumer. This feature offers information on the user's prior interactions with the system.

**Integration**

These features enable the system to keep an extensive customer record, which makes a major contribution to customer relationship management. Personalised engagements and overall effectiveness are improved with the opportunity to add new customers, see past consumer details, amend information, and check purchase history.

1. **Point of Sale (POS) Main Menu**

To manage the sales transaction process within the Grocery Management System, the Point-of-Sale System is essential. The primary choices consist of:

Functions

* *Add to Cart (`add\_to\_cart`):* Incorporates items into a customer's purchasing basket. This feature replicates the procedure of a consumer choosing products to buy.
* *View Cart (`view\_cart`):* Shows the purchased cart's contents as they are at the moment. A real-time summary of the products the consumer has chosen is provided by this feature.
* *Checkout (`checkout`):* Gives the entire cost and associates the purchase with a consumer, simulating the checkout process. This feature refreshes the purchase history and completes the transaction.

**Integration**

These elements guarantee a seamless and effective transaction by simulating the actual checkout experience. Overall efficiency and usability are improved by the system's capacity to compute the entire cost, link purchases to consumers, and display a clear summary of the shopping cart.

1. **Inventory Control Main Menu**

The Grocery Management System can successfully manage stock levels with the use of inventory control's vital tools. The primary choices consist of:

*Functions*

* *View Stock (`view\_stocks`):* Shows an extensive view of the available goods, along with pricing and volume. The entire inventory state is shown in a snapshot via this function.
* *Receive Stock (`receive\_stock`): Enables users to receive more of a particular product to revise the amount of stock. This feature makes refilling stock easier.*
* *Set Low-Stock Alert (`set\_low\_stock\_alert`):* constantly manages inventory by establishing a threshold for low-stock warnings. This feature improves the system's capacity to avoid shortages of products.

**Integration**

This group of operations guarantees that the stock is kept in good condition and is refilled when necessary. Proactive inventory management is improved by the ability to inspect stock, get additional quantities, and establish low-stock alerts. These features also improve overall efficiency and usability.

**Testing and Quality**

Extensive testing was conducted on the Grocery Management System to guarantee precision, efficiency, and user-friendliness. To guarantee a proper presentation of the inventory, the View Items functionality—which is controlled by view\_items—went through testing. To verify that the function consistently presents an orderly view of the inventory, a variety of inputs were used in the test. The unit test made sure that new items could be successfully added to the inventory for the Add Product functionality, which is controlled by add\_product. To verify the function's dependability, several scenarios were simulated, including various product names, volumes, and prices. Product details had to be changed to test the Update Product function.

The careful unit testing procedure included functions like remove\_product, purchase\_product, view\_order, update\_order, view\_purchase\_history, add\_cart. This extensive testing contributes to the overall quality and dependability of the Grocery Management System by ensuring its resilience and dependability.

**Future Work**

* **GUI Interface using Tkinter:** From Command Line to GUI-based Grocery Management System with UX/UI experience design.
* **User Authentication System:** By using user authentication, adding login and logout features with the owner.
* **Database Integration (MySQL):** In this Command line-based system there is only a list or dictionaries used but to make our application powerful a database is a must locally or cloud-based.

**Conclusion**

In conclusion, the creation of the Grocery Management System is evidence of how practical solutions and technology innovation have come together in the grocery retail industry. Python's readability, ease of use, and wide library support made it the perfect programming language for creating a flexible and intuitive application. Functionality, command line interface, and integration issues were carefully handled during the testing process, highlighting the vital significance that thorough testing plays in guaranteeing the system's dependability. Debugging approaches were essential in locating and fixing problems, which enhanced the final application's resilience.