DE LA SALLE UNIVERSITY - MANILA

STOCKIFY: Inventory Management System

A Term Project
Presented to Mr. Ramon Stephen Ruiz
In Partial Fulfillment of the
Requirements for the Course Programming Logic and Design (PROLOGI)

By:

Bandigan, Dustin Josh R. - DJRB

Yang, Yuankai - YY
EQ3
M-TH 12:45 - 1:45
April, 2023

TABLE OF CONTENTS

I. INTRODUCTION

Marketing is a vital component of any business strategy, as it helps organizations reach and engage with their target audience. With the constant development of new technology, tools, and channels in today's rapidly changing business world, marketing has significantly evolved. As a result, marketing system management has become a key area of focus for businesses hoping to stay competitive and relevant in their own industries. The market has become more competitive as a result of this growth, and businesses are continuously looking for new strategies to set themselves apart from the competition.

Our project, namely: “STOCKIFY: Inventory Management System” is an inventory management system that can help manage inventory of items that are wished to be stored. This program includes User Interface for inputting specific information of the items, Stores the quantity and prices of the stored items, upon purchasing, it totals the items with the set unit prices and totals it to its final amount.

A. Background of the Study

Stock management is a crucial part of any business, no matter how big or small. The effective administration of stock assists with guaranteeing that a business can fulfill its client need while downplaying functional expenses. Over- or under-supply can result from inaccurate or ineffective stock management, which can cause a company to incur significant financial losses.

Stock management has traditionally been performed manually, which can be costly, error-prone, and time-consuming. A stock management system, on the other hand, enables businesses to now automate their stock management procedures thanks to technological advancements. A software program called a stock management system is made to keep track of orders, sales, deliveries, and inventory levels. It reduces the likelihood of stockouts and overstocking, streamlines ordering procedures, and enables businesses to monitor stock levels in real time.

There are numerous advantages to implementing a stock management system, including cost reduction, increased efficiency, and increased profitability. However, not all companies have implemented stock management systems, and some continue to struggle with manual procedures. As a result, it is necessary to comprehend the factors that influence businesses' adoption of stock management systems.

The goal of this study is to find out what factors influence businesses' adoption of stock management systems. It will investigate the advantages and difficulties of utilizing a stock administration framework and assess its effect on business execution. This study's findings will shed light on the considerations that businesses need to make when implementing a stock management system, allowing them to make more educated choices about this technology.

B. Problem Statement

In the present serious business climate, proficient stock administration is basic for the outcome of any business. However, manual stock management procedures are still used by many businesses, and they can be time-consuming, costly, and prone to errors. Stockouts, overstocking, and inaccurate inventory records caused by a lack of an automated stock management system can lower customer satisfaction, increase costs, and reduce profitability.

Not all businesses have adopted stock management systems despite their numerous advantages, including increased efficiency, decreased costs, and improved accuracy. Due to the perceived complexity and potential disruption to their operations, some businesses may be reluctant to implement a stock management system because it requires a significant investment of time and resources.

In addition, it is unclear what factors influence businesses' adoption of stock management systems. As a result, it is necessary to investigate both the obstacles businesses face when implementing stock management systems and the factors that encourage their adoption. This study expects to address this hole in the writing by investigating the variables that impact the reception of stock administration frameworks in organizations and assessing the effect of this innovation on business execution. Businesses can learn about the advantages of implementing this technology to enhance their stock management procedures from the study's findings, which can shed light on the obstacles to their use.

C. **Objectives**

C.1 General Objective

Our primary objective is the creation of a tool that preserves prices and quantities while simultaneously generating a list of stored items. Users will be able to track the quantity of their products by using this program. The features of the program will include a list of products and a receipt for the customer's purchases.

C.2 Specific Objectives

- Helping staffs take inventory of various items so they can decide when to replenish.
- Provide customers with purchasing advice
- Clear receipts for customers to pay and staff to change money

D. **Significance of the Study**

Businesses, researchers, and policymakers can all benefit from this study's findings in a number of ways. First and foremost, the study can assist businesses in comprehending the advantages of implementing a stock management system and the factors that influence its adoption. This can help businesses decide whether to implement a stock management system and what factors to take into account when choosing one.

Also, the review can furnish policymakers with bits of knowledge into the difficulties that organizations face while embracing stock administration frameworks and the advantages that organizations can get from this innovation. This can help policymakers understand the need for policies and incentives to encourage businesses to use stock management systems, which can make the business sector more efficient and competitive.

Finally, by examining the effect of this technology on business performance, the research can add to the existing body of knowledge regarding stock management systems. Future research on this subject can be informed by the findings of this study, which can provide researchers with insights into the advantages and drawbacks of stock management systems.

In conclusion, this research has the potential to enhance our comprehension of stock management systems and the effects they have on businesses. This study's findings can help businesses, policymakers, and researchers make well-informed decisions regarding the adoption and implementation of stock management systems, which can boost competitiveness and performance.

II. Review of Related Literature

The following literature review focuses on the factors that influence the adoption of stock management systems and the significance of stock management systems for businesses.

1. The Importance of Systems for Stock Management:

In today's market, stock management systems have become increasingly important for businesses to keep their edge over competitors. Bakker et al. say that (2016), stock management systems can help businesses reduce costs, increase customer satisfaction, and optimize their inventory levels. Additionally, Hasan et al.'s research (2017) found that stock administration frameworks can work on the exactness of stock records, diminish stockouts and overloading, and increment the effectiveness of request satisfaction.

2. The following factors influence the use of stock management systems:

The adoption of stock management systems in businesses can be influenced by a number of factors. As indicated by Brown et al. (2015), these variables incorporate the apparent helpfulness of the innovation, the similarity of the framework with the current business processes, the convenience of the framework, and the accessibility of assets to carry out and keep up with the framework. Research by Ahmed et al. (In addition, the perceived complexity of the system, the expense of implementation, and resistance to change were identified as barriers to stock management system adoption.

3. How Stock Management Systems Affect Business Results:

It has been found that businesses perform better when stock management systems are used. Kaur et al.'s study says that (2017), stock administration frameworks can prompt expanded deals, further developed consumer loyalty, and decreased functional expenses. Moreover, a concentrate by Ojah et al. (2017) discovered that stock management systems can make the supply chain more efficient and shorten lead times, making businesses more adaptable and competitive.

Generally speaking, the writing audit features the significance of stock administration frameworks in organizations and the variables that impact their reception. The implementation of a stock management system can improve business performance and competitiveness by reducing costs, enhancing efficiency, and improving inventory management.

III. Methodology

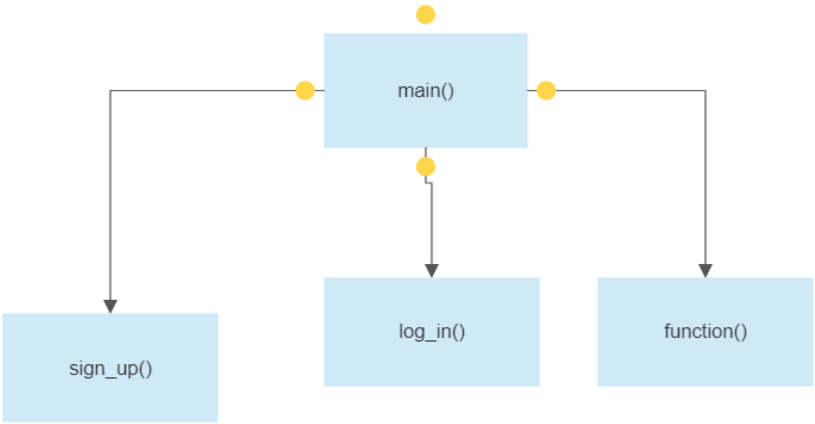
A. Conceptual Framework - IPO Chart (Input - Process - Output)

Input	**Process**	**Output**
The client is given a menu and provoked to pick a choice. - Sign up - Log in - Quit If option 1 is chosen, the user will be asked to	User selects an option from the menu If the user selects option 1, the program will prompt the user to enter a username and password. If the username already exists, the user will be asked to choose a different	If the user selects option 1, the program will create a new account and display a message indicating that the account was created successfully. If the user selects option 2 and successfully logs in, the program will

enter a username and password. They will be prompted to select a different username if the username already exists. They will be asked to try again if the password and confirm password fields do not match. The user will be prompted to enter their username and password if they select option 2. In the event that the username doesn't exist or the secret phrase is mistaken, they will be approached to attempt in the future. An inventory system will be displayed to the user if the login is successful. The client will be provoked to enter the name, unit cost, and amount of things to be added to the stock. Until the user decides to stop, the program will keep asking for more items to be added. The item's name, unit price, and quantity will be listed in the inventory list. After that, the user will be prompted to enter the quantity and serial number of the item they wish to purchase. The program will display an error message and prompt the user to try again if the quantity	username. If the password and confirm password fields do not match, the user will be asked to try again. If the user selects option 2, the program will prompt the user to enter their username and password. If the username does not exist or the password is incorrect, the user will be asked to try again. If the user successfully logs in, the program will present the user with an inventory system. The user will be prompted to enter the name, unit price, and quantity of items to be added to the inventory. The program will keep asking for more items to be added until the user chooses to stop. The inventory list will be displayed with the serial number, item name, unit price, and quantity. The user will be prompted to enter the serial number of the item they want to purchase and the quantity they want to buy. If the quantity requested is greater than the available quantity, the program will display an error message and ask the user to try again. The program will continue to	display a message indicating that the login was successful. The program will display the inventory list, receipt

requested is greater than the quantity available. Until the user decides to stop, the program will keep asking for more purchases. The program will show the user a receipt with the item name, unit price, quantity, and total price of each item purchased after the user has finished making purchases. The all out cost, installment sum, and change will likewise be shown. The remaining quantity of each item will be displayed in an updated inventory list at the end. Assuming the client chooses choice 3, the program will stop.	ask for more items to be purchased until the user chooses to stop. Once the user is done purchasing items, the program will display a receipt showing the item name, unit price, quantity, and total price of each item purchased. The total price, payment amount, and change will also be displayed. Finally, an updated inventory list will be displayed with the remaining quantity of each item. If the user selects option 3, the program will quit.	

B. Hierarchy Chart

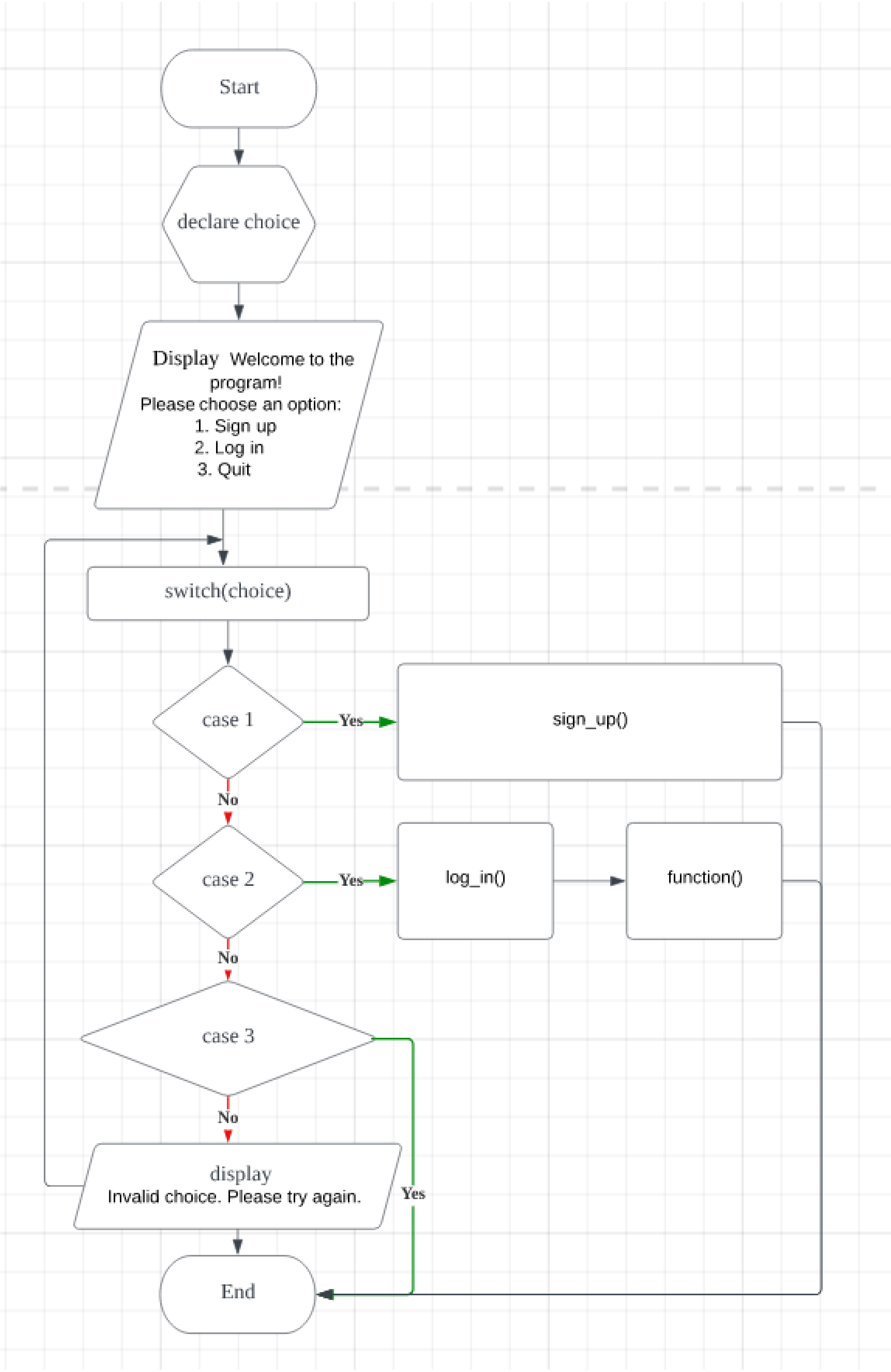


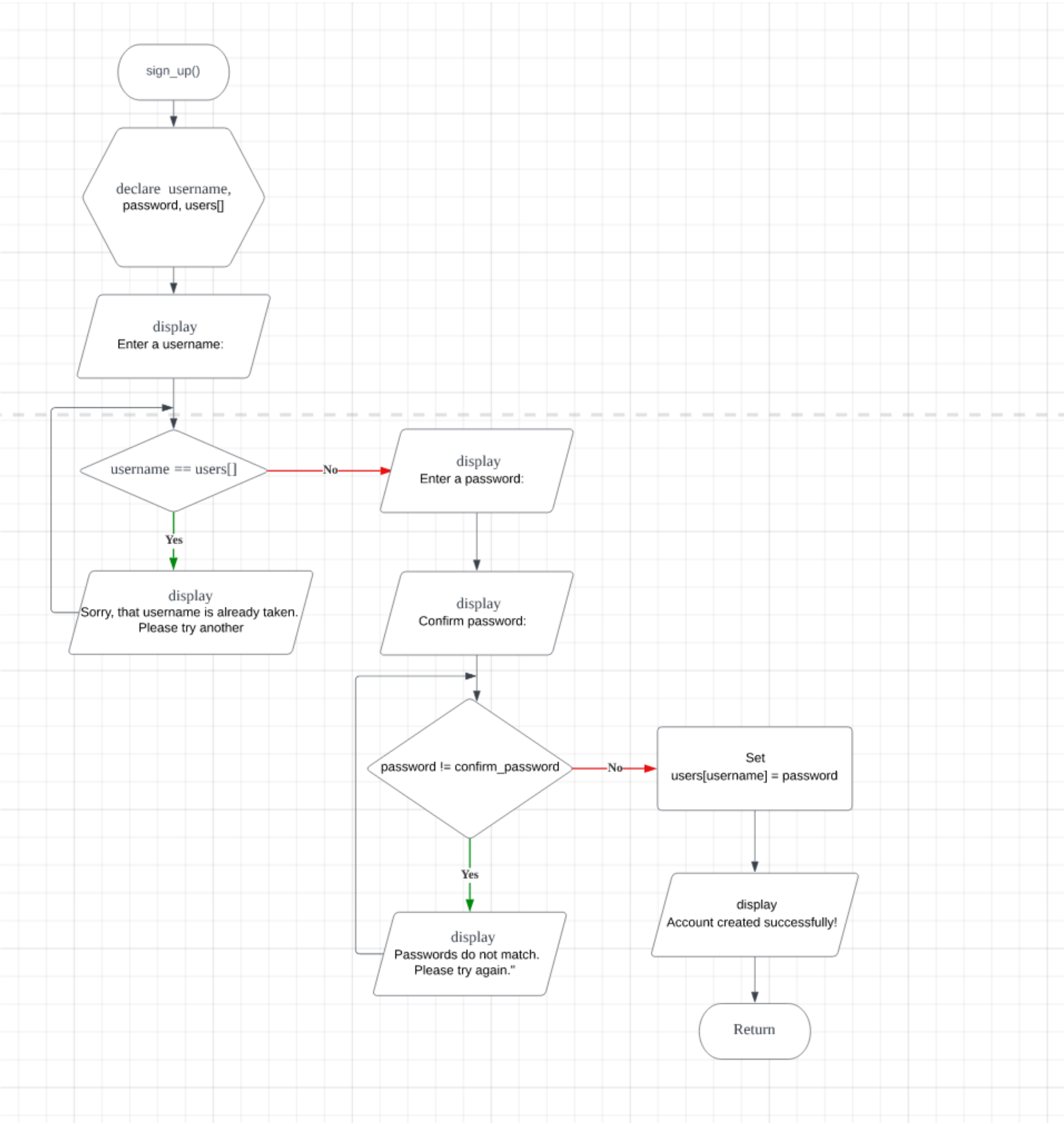
C. Flowchart

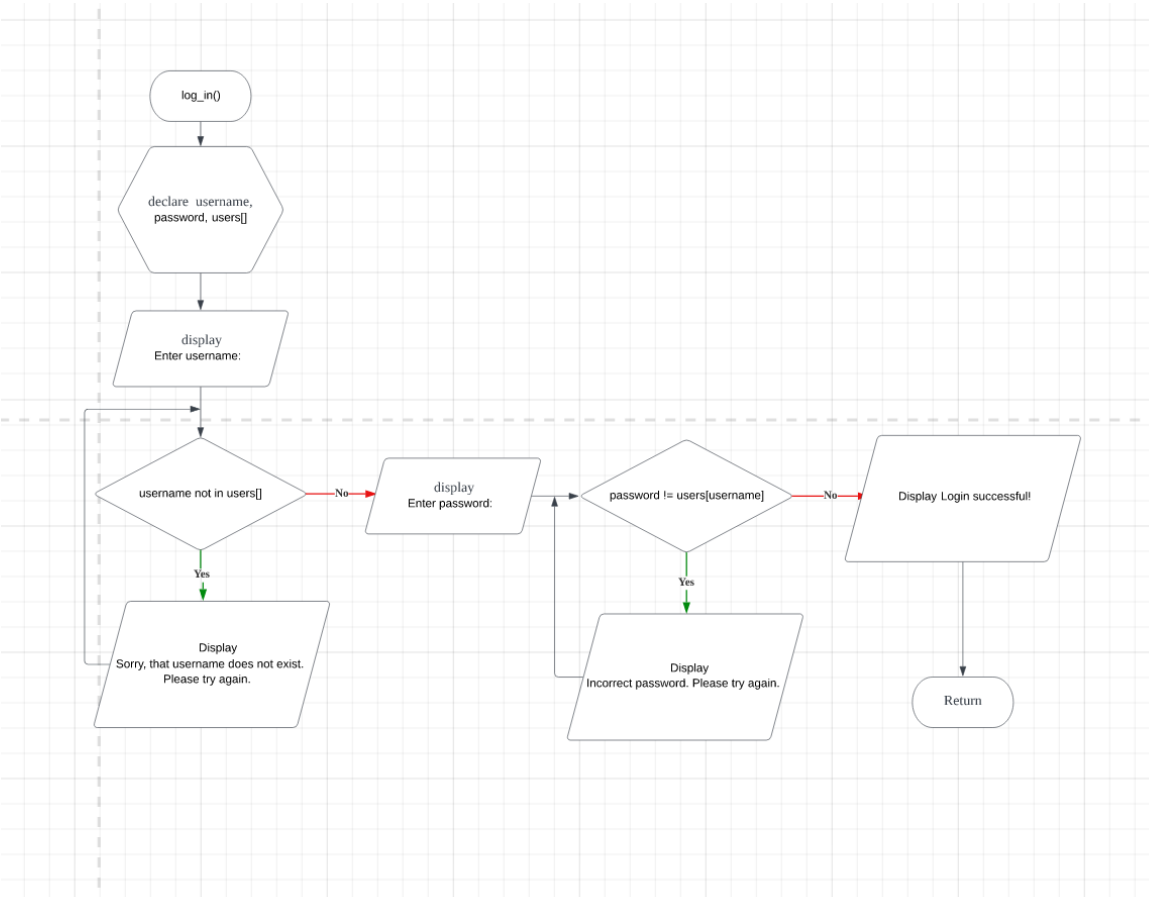
To view the flowchart in full quality, refer to the link below: (Note: Open it through lucid.app)

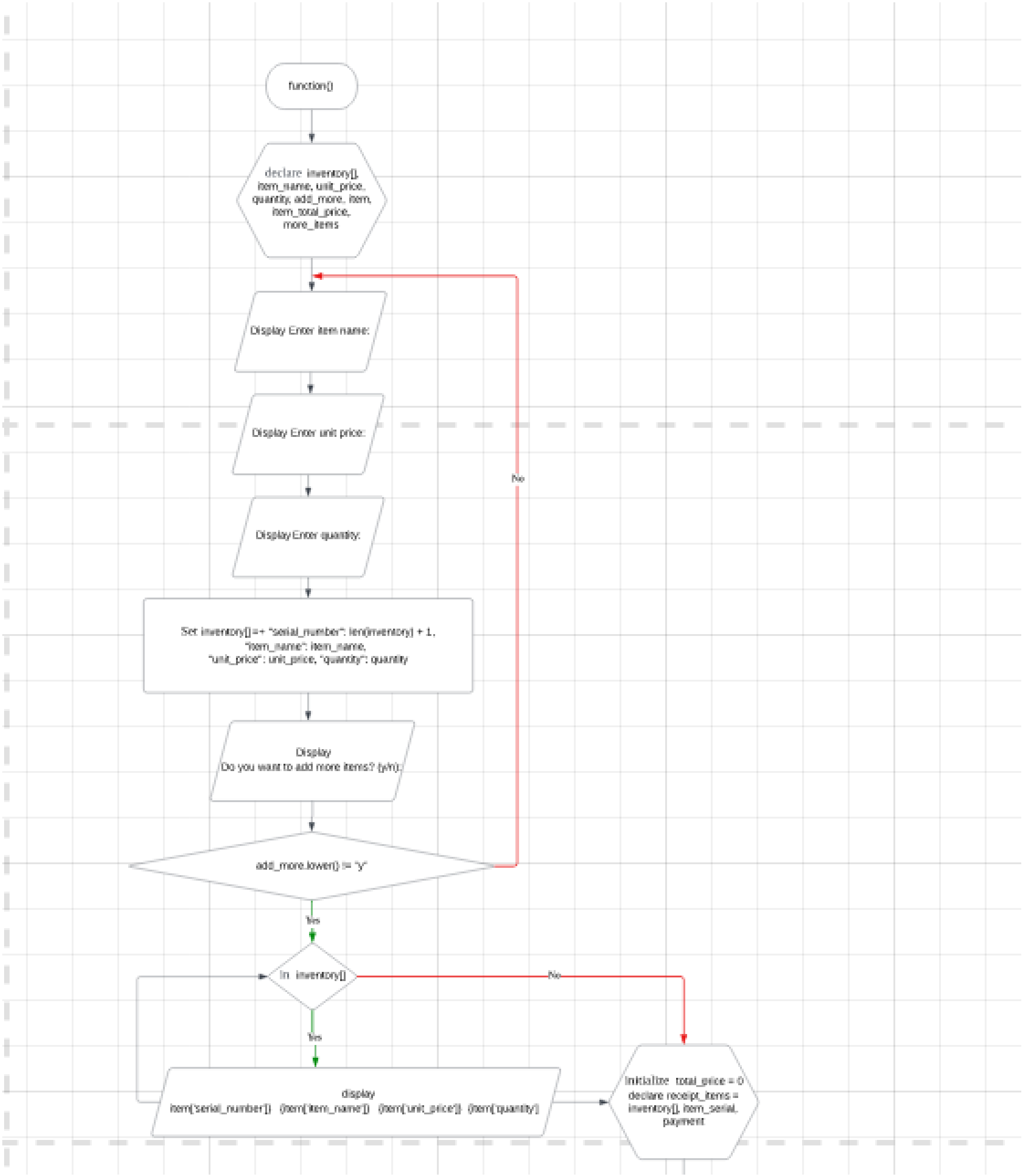
TeamZeus Flowchart

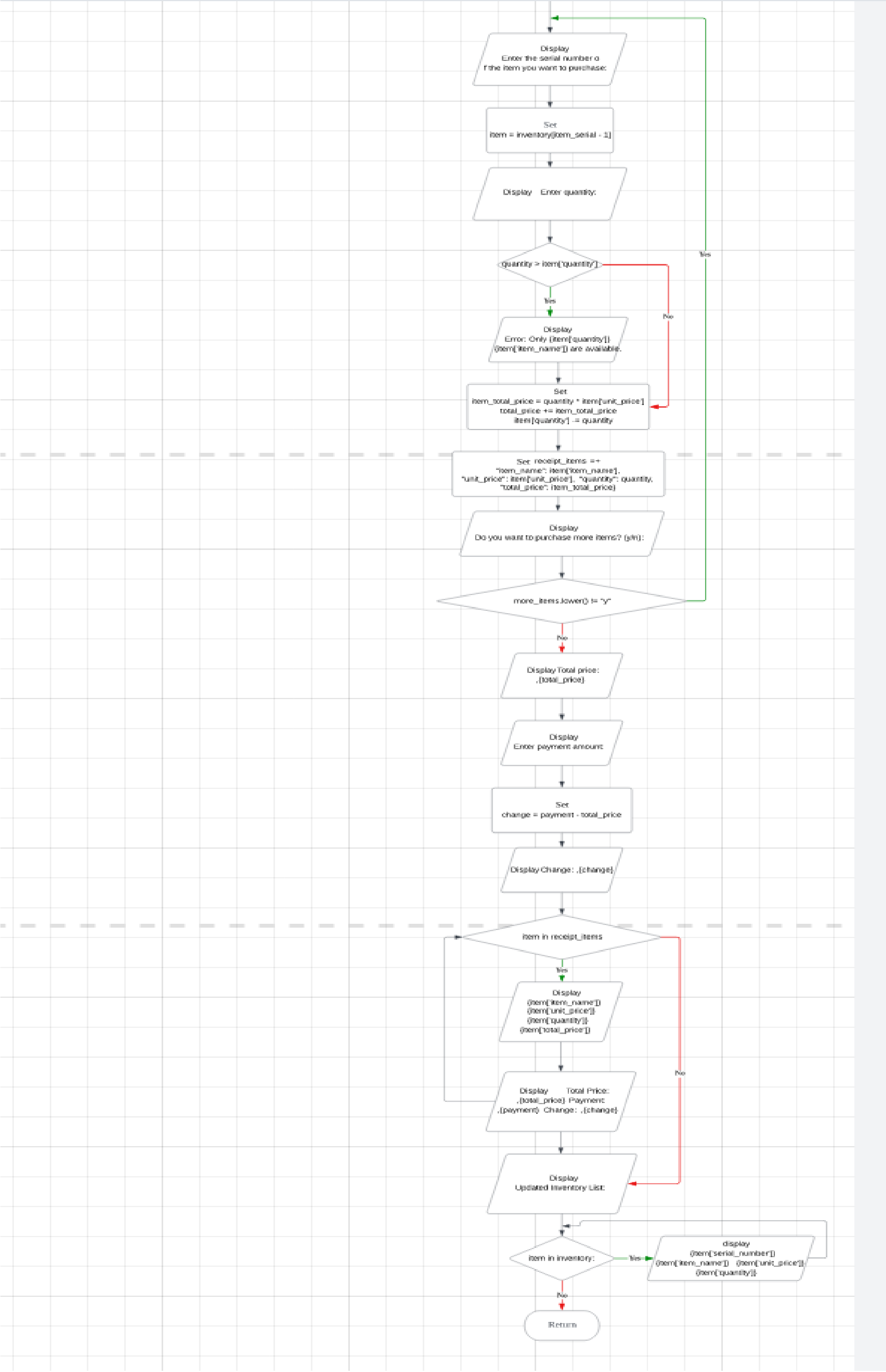
Full view of the Flowchart:


Start
declare choice
Display Welcome to the program!
Please choose an option:
1. Sign up
2. Log in
3. Quit
switch(choice)
case 1
Yes
sign_up()
No
case 2
Yes
log_in()
function()
No
case 3
No
display
Invalid choice. Please try again.
Yes
End


sign_up()
declare username,
password, users[]
display
Enter a username:
username == users[]
No
Yes
display
Sorry, that username is already taken.
Please try another
display
Enter a password:
display
Confirm password:
password != confirm_password
No
Yes
Set
users[username] = password
display
Passwords do not match.
Please try again."
display
Account created successfully!
Return


log_in()
declare username, password, users[]
display Enter username:
username not in users[]
No
Yes
Display Sorry, that username does not exist. Please try again.
display Enter password:
password != users[username]
No
Yes
Display Incorrect password. Please try again.
Display Login successful!
Return


function()
declare inventory[], item_name, unit_price, quantity, add_more, item, item_total_price, more_items
Display Enter item name:
Display Enter unit price:
Display Enter quantity:
Set inventory[] =+ "serial_number": len(inventory) + 1, "item_name": item_name, "unit_price": unit_price, "quantity": quantity
Display
Do you want to add more items? (y/n):
add_more.lower() != "y"
No
Yes
In inventory[]
No
Yes
display
item['serial_number'] (item['item_name']) (item['unit_price']) (item['quantity']
Initialize total_price = 0
declare receipt_items = inventory[], item_serial, payment


Display
Enter the serial number o
f the item you want to purchase:
Set
item = inventory[item_serial - 1]
Display Enter quantity:
quantity > item['quantity']
Yes
No
Display
Error: Only {item['quantity']}
{item['item_name']} are available.
Set
item_total_price = quantity * item['unit_price']
total_price += item_total_price
item['quantity'] -= quantity
Set receipt_items =+
"item_name": item['item_name'],
"unit_price" : item['unit_price'], "quantity": quantity,
"total_price": item_total_price}
Display
Do you want to purchase more items? (y/n):
more_items.lower() != "y"
Yes
No
Display Total price:
,{total_price}
Display
Enter payment amount:
Set
change = payment - total_price
Display Change: ,{change}
item in receipt_items
Yes
Display
{item['item_name']}
{item['unit_price']}
{item['quantity']}
{item['total_price']}
Display Total Price:
,{total_price} Payment:
,{payment} Change: ,{change}
No
Display
Updated Inventory List:
item in inventory:
Yes
display
{item['serial_number']}
{item['item_name']} {item['unit_price']}
{item['quantity']}
No
Return

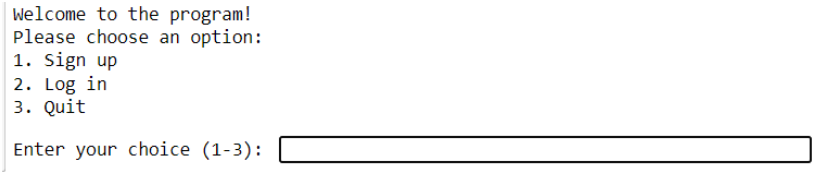
D. Pseudocode

```
Module sign_up()
    Declare username, password, users[]
    Display Enter a username:
    Input username
    IF username == users[], THEN
        Display Sorry, that username is already taken.  Please try another
    ELSE
        Display Enter a password:
        Input password
        Display Confirm password:
        IF password != confirm_password, THEN
            Display Passwords do not match.  Please try again."
        ELSE
            Set users[username] = password
            Display Account created successfully!
        END IF
    END IF
END Module
```

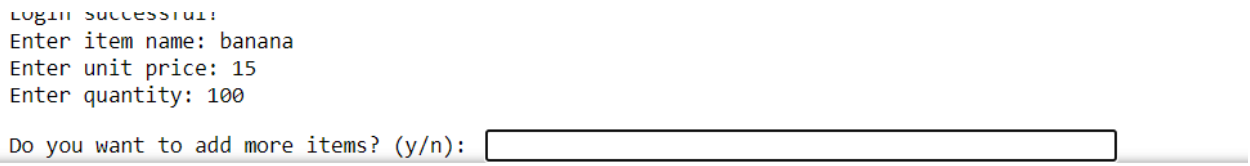
```
Module log_in()
    Declare username, password, users[]
    Display Enter username:
    Input username
    IF username not in users[], THEN
         Display Sorry, that username does not exist.  Please try again.
    ELSE
        Display Enter password:
        Input password
        IF password != users[username], THEN
             Display Incorrect password. Please try again.
        ELSE
            Display Login successful!
        END IF
    END IF
END Module
```

```
Module function()
    Declare inventory[], item_name, unit_price, quantity, add_more, item, item_total_price,
    more_items
    Display Enter item name:
    Input item_name
    Display Enter unit price:
    Input unit_price
    Display Enter quantity:
    Input quantity
    Set inventory[] =+ "serial_number": len(inventory) + 1,  "item_name": item_name,
    "unit_price": unit_price, "quantity": quantity
    Display Do you want to add more items? (y/n):
    Input add_more
    IF add_more.lower() != "y", THEN
        IF in inventory[], THEN
            Display item['serial_number']}  {item['item_name']}  {item['unit_price']}
            {item['quantity']
        ELSE
            initialize total_price = 0 declare receipt_items = inventory[], item_serial,
            payment
            Display Enter the serial number o f the item you want to purchase:
            Input  item_serial
            Set item = inventory[item_serial - 1]
            Display Enter quantity:
            Input quantity
            IF quantity > item['quantity'], THEN
                Display Error: Only {item['quantity']}   {item['item_name']} are
                available
            Else
                Set
                item_total_price = quantity * item['unit_price']
                total_price += item_total_price
                item['quantity'] -= quantity
                Set
                receipt_items =+ "item_name": item['item_name'],  "unit_price":
                item['unit_price'],  "quantity": quantity,  "total_price":
                item_total_price}
                Display Do you want to purchase more items? (y/n):
                IF more_items.lower() != "y", THEN
                    Go back to Display Enter the serial number o f the item you
                    want to purchase:
                ELSE
                    Display Total price: ,{total_price}
                    Display Enter payment amount:
                    Input payment
                    Set change = payment - total_price
                    Display Change: ,{change}
                    IF item in receipt_items, THEN

                        Display {item['item_name']} {item['unit_price']}
                        {item['quantity']} {item['total_price']}
                        Go back to IF item in receipt_items
                    ELSE
                        Display Updated Inventory List:
                        Display {item['serial_number']}
                        {item['item_name']} {item['unit_price']}
                        {item['quantity']}
                    END IF
                END IF
            END IF
        END IF

    ELSE
        Go back to  Display Enter item name:
    END IF
END Module

Module main()
    Declare choice
    Display Welcome to the program!
         Please choose an option:
         1. Sign up
         2. Log in
         3. Quit
    Input choice
    IF choice == 1, THEN
        Call sign_up()
    ELSE IF choice == 2, THEN
        Call log_in()
            Call function()
    ELSE IF choice == 2, THEN
        END Module
    ElSE
        display Invalid choice. Please try again.
        Go back to Input choice
    END IF
END Module
```

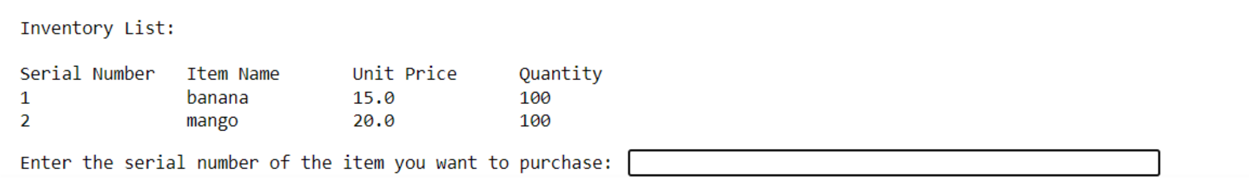
IV. Result



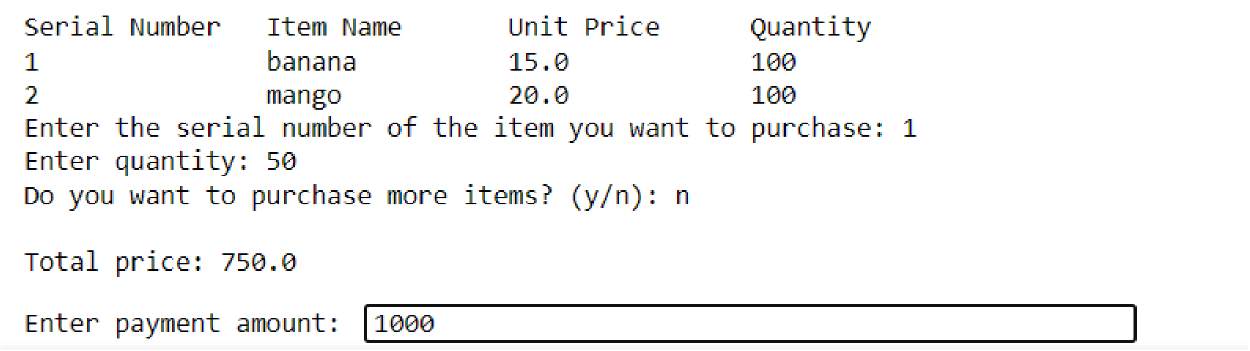
```
Welcome to the program!
Please choose an option:
1. Sign up
2. Log in
3. Quit

Enter your choice (1-3):
```



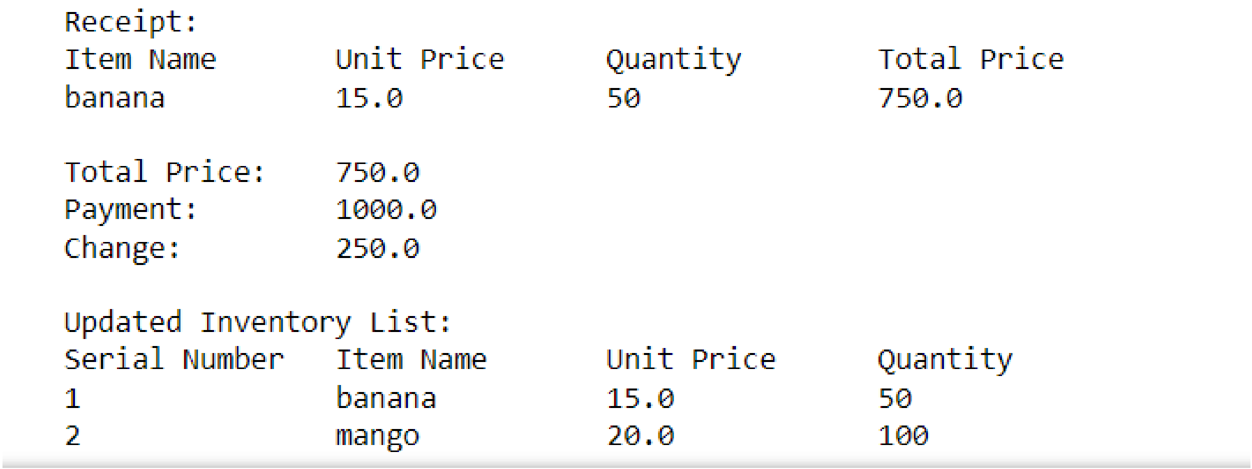
```
Login successful!
Enter item name: banana
Enter unit price: 15
Enter quantity: 100

Do you want to add more items? (y/n):
```



```
Inventory List:

Serial Number   Item Name      Unit Price     Quantity
1               banana         15.0           100
2               mango          20.0           100

Enter the serial number of the item you want to purchase:
```



```
Serial Number   Item Name      Unit Price     Quantity
1               banana         15.0           100
2               mango          20.0           100
Enter the serial number of the item you want to purchase: 1
Enter quantity: 50
Do you want to purchase more items? (y/n): n

Total price: 750.0

Enter payment amount: 1000
```



```
Receipt:
Item Name       Unit Price      Quantity        Total Price
banana          15.0            50              750.0

Total Price:    750.0
Payment:        1000.0
Change:         250.0

Updated Inventory List:
Serial Number   Item Name       Unit Price      Quantity
1               banana          15.0            50
2               mango           20.0            100
```

V. Discussion of Results

To protect user privacy and data, the program offers a secure sign-up and login system. During the login process, the program can quickly and easily authenticate user credentials by using a dictionary to store usernames and passwords.

Users can add items to an inventory list using the inventory management system. A dictionary stores the name, unit price, and quantity of each item. A point-of-sale system that lets users purchase items from the inventory list is also included in the program. The program calculates the total price of the purchase, which is displayed on the receipt, and verifies the availability of an item based on its stock quantity.

Overall, the program works and has useful features for managing sales transactions and inventory. However, additional features like data visualization for inventory management and sales analytics may be added to this program in the future to provide additional insights into business operations. Also, carrying out safety efforts to safeguard client information, like encryption or two-factor verification, could improve the program's security and protection highlights.

VI. Analysis, Conclusion, and Future Directives

Analysis:

An easy-to-use inventory management system that lets users create an account, log in, add items to the inventory, and purchase them is provided by the Python program that is presented. A dictionary is used to store user information like usernames and passwords, and a list of dictionaries is used to store inventory information like item names, unit prices, and quantities. Functions for creating an account, logging in, and inventory management are used to structure the program. Additionally, it makes use of loops for data processing and user input.

The program begins by showing a menu that permits clients to make a record, sign in, or quit the program. The user is prompted to enter a unique username and password if they decide to create an account. The application will prompt the user to try again if the username is already taken. The user is prompted to try again if the password and confirm password fields do not match. When the record is effectively made, the program adds the username and secret word to the clients word reference.

The user is prompted to enter their username and password if they choose to log in. The program checks if the username and secret key are right by finding them in the clients word reference. If the username or secret word is mistaken, the client is approached to attempt once more. The program moves on to the inventory management system if the login is successful.

The stock administration framework permits clients to add things to the stock and buy things from the stock. The name, unit price, and quantity of an item are entered by the user when they add it to the inventory. This information is added to a dictionary by the program and appended to the inventory list. The program then, at that point, inquires as to whether the client needs to add more things. If the user decides to buy something, they have to enter the item's serial number and the quantity they want to buy. The program determines whether the item is in stock and whether there is sufficient supply. The program subtracts the quantity from the inventory and calculates the total price if there is sufficient quantity. The program then inquires as to whether the client needs to buy more things. The program displays the total price, prompts the user to enter the payment amount, calculates the change, and displays a receipt after the user has completed the purchase.

Conclusion:

The stock administration framework introduced is a basic yet viable answer for independent ventures or people who need to monitor their stock. The program is not difficult to utilize and gives a direct method for adding and buying things from the stock. Additionally, the program stores user and inventory data by making use of fundamental data structures like dictionaries and lists. People who want to learn the fundamentals of Python programming or small businesses that require a straightforward inventory management system may find this application useful.

Future Orientations:

The program could be improved or expanded in a number of areas. To begin, the program could be altered to make it possible to edit or delete items from the inventory. Users would have more control over how they manage their inventory as a result of this. Second, allowing multiple users to access the inventory system could be added to the program. Adding user authentication and access control mechanisms to the program is necessary for this. Lastly, the program could be altered to include additional functions like report generation or sales data tracking. These elements would give clients more prominent experiences into their stock and deals execution. This inventory management system has a lot of room for growth and improvement in the future.

VII. References

Bakker, E., Verbraeck, A., & Van Donk, D. (2016). The potential of inventory management through advanced demand information. International Journal of Production Economics, 181, 242-252. doi: 10.1016/j.ijpe.2016.08.016

Hasan, S., Khan, S. U., & Ahmad, R. (2017). Inventory management through vendor managed inventory (VMI) for a two-level supply chain with deteriorating items. Computers & Industrial Engineering, 105, 215-227. doi: 10.1016/j.cie.2017.01.015

Brown, S. A., Massey, A. P., Montoya-Weiss, M. M., & Burkman, J. R. (2015). Do I really have to? User acceptance of mandated technology. European Journal of Information Systems, 24(4), 367-382. doi: 10.1057/ejis.2014.17

Ahmed, A. M., Islam, M. A., & Rahman, M. M. (2018). Investigating the determinants of RFID adoption in inventory management: Evidence from Bangladeshi firms. Journal of Business Research, 88, 360-368. doi: 10.1016/j.jbusres.2018.01.042

Kaur, H., Singh, H., & Batra, R. (2017). Evaluating the impact of inventory management on business performance: Evidence from Indian manufacturing firms. Global Business Review, 18(2), 460-471. doi: 10.1177/0972150916683909

Ojah, K., Isa, D., Yusuf, A., & Aremu, A. (2017). The impact of supply chain management on business performance: A case study of Nigerian oil and gas industry. Journal of Manufacturing Technology Management, 28(3), 374-397. doi: 10.1108/JMTM-05-2016-0069

VIII. Appendices

A. User's Manual

A simple inventory system with user authentication is implemented in this Python program. Users can register, log in using a username and password, create an inventory list, and then purchase items from the inventory.

There are three choices on the program's menu: sign up, sign in, and quit. The client can choose a choice by entering a number from 1 to 3.

1. Sign Up: A new user can set up an account with this option by entering a unique username and password. If the username is already taken, the program will check to see if it is available and prompt the user to try a different one. The user will be prompted to enter a password and confirm the password if the username is available. The program will create a new account and store the username and password in a dictionary if the two passwords match.

2. Log in: Existing users can log in using their username and password with this option. The username and password will be compared to those in the dictionary by the program. The user can access the inventory system if they match. The application will prompt the user to try again if they do not match.
3. Quit: This choice permits the client to leave the program.

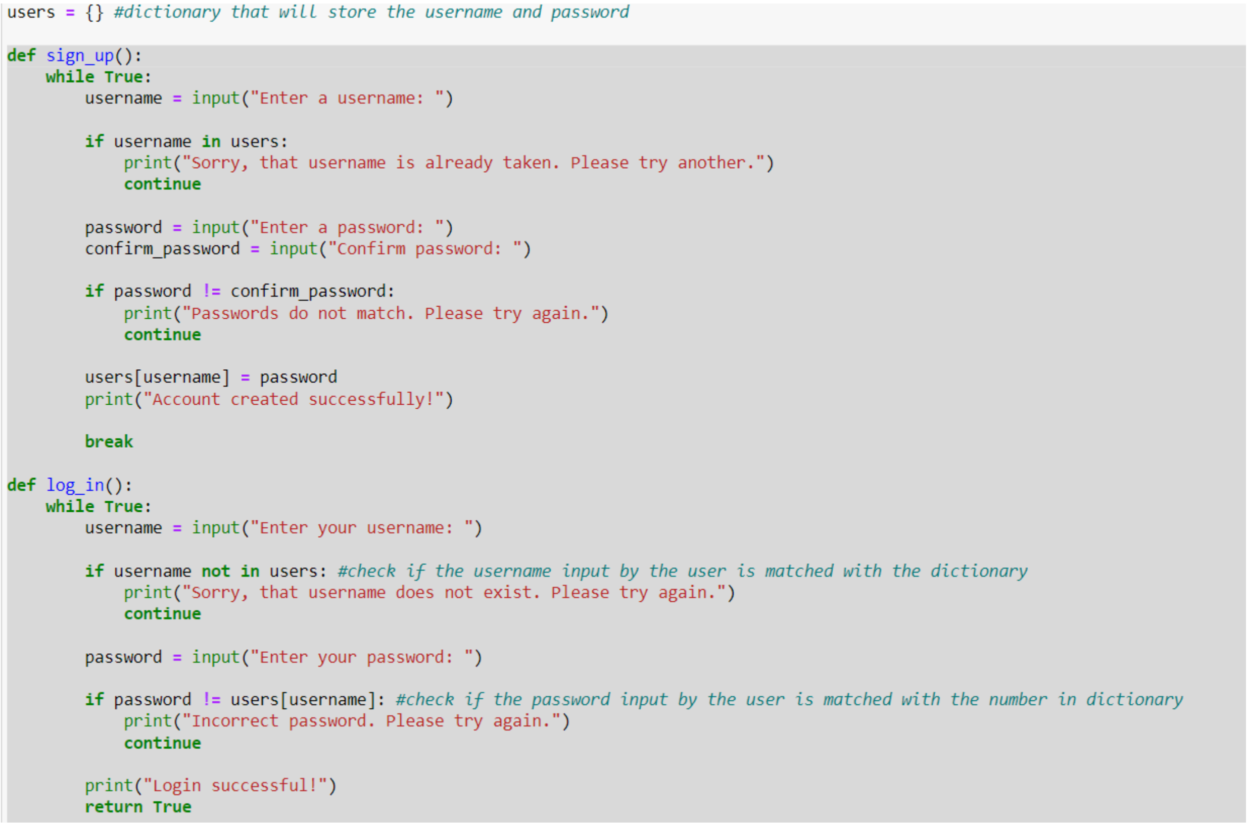
The user can enter the item's name, unit price, and quantity to create an inventory list after logging in. Each item will be added to a list of dictionaries that hold the item's information by the program. The user can either quit or add more items to the list.

The user can make purchases from the inventory list by entering the item's serial number and quantity once the list has been created. The program will check to see if the item is available and figure out how much it will cost all together. The user can keep using or buy more items. The program will print a receipt with the list of items purchased, total price, payment amount, and change after the user has completed the inventory system. Additionally, it will print the most recent inventory list, which includes the updated quantities of each item.

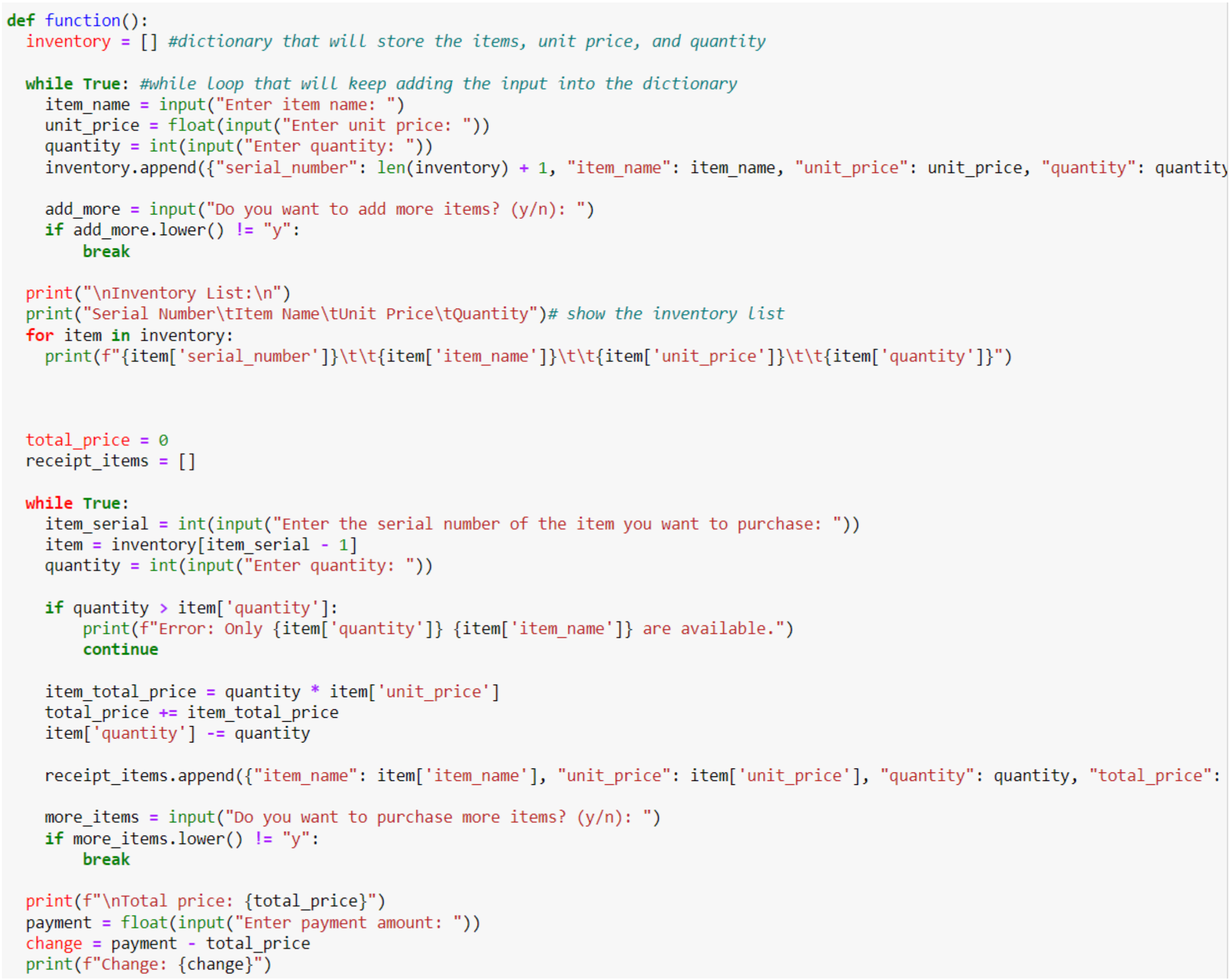
Follow these steps to use the program:
1. Open a command prompt or terminal.
2. Explore to the index where the Python program is found.
3. Follow the name of the program file (including the ".py" extension) by typing "python."
4. To use the program, follow the on-screen instructions.

Note: You should have Python introduced on your PC to run the program. On the off chance that you don't have Python introduced, you can download it from the authority site: https://www.python.org/downloads/

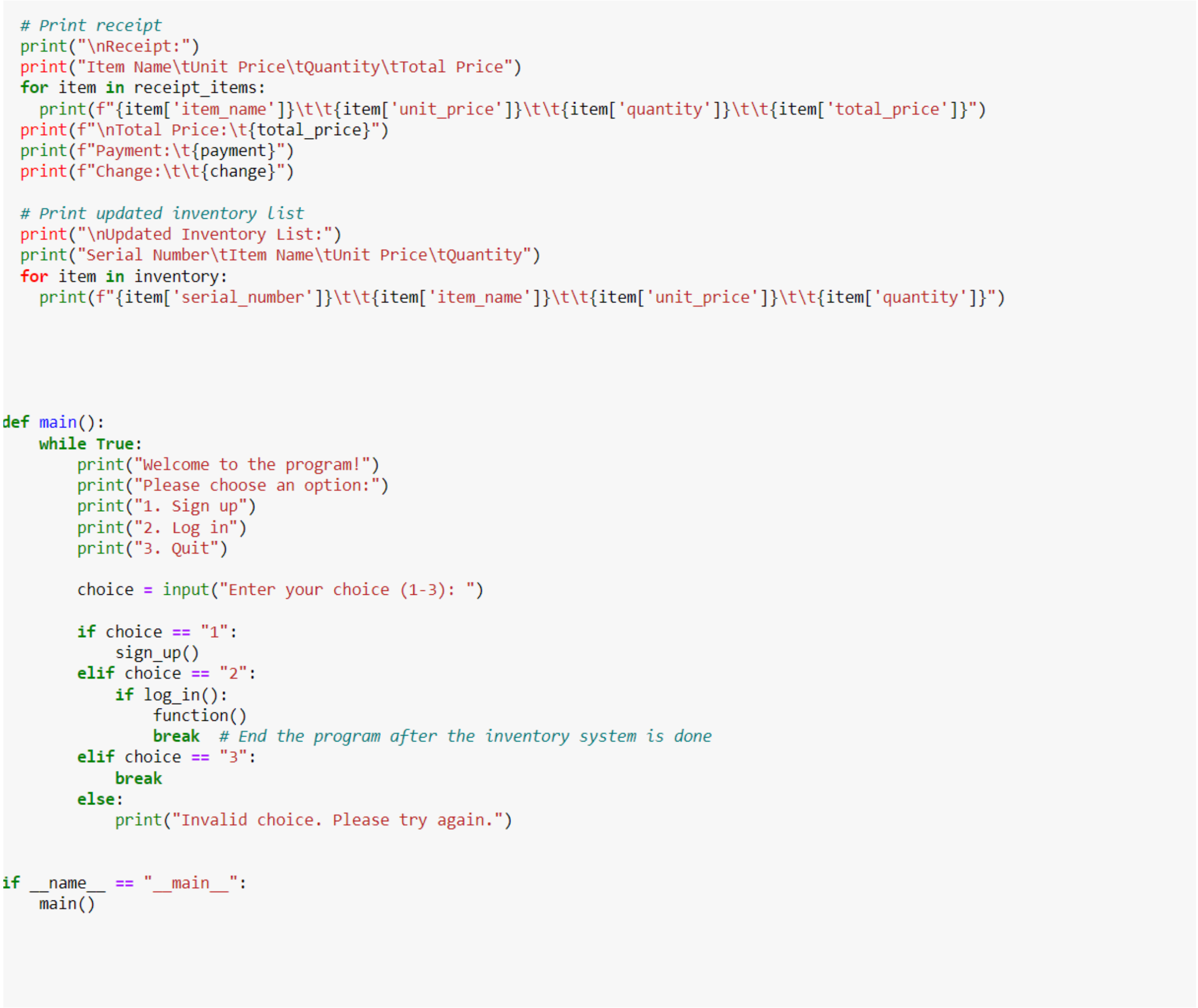
B. Source Code



```python
users = {} #dictionary that will store the username and password

def sign_up():
    while True:
        username = input("Enter a username: ")

        if username in users:
            print("Sorry, that username is already taken. Please try another.")
            continue

        password = input("Enter a password: ")
        confirm_password = input("Confirm password: ")

        if password != confirm_password:
            print("Passwords do not match. Please try again.")
            continue

        users[username] = password
        print("Account created successfully!")

        break

def log_in():
    while True:
        username = input("Enter your username: ")

        if username not in users: #check if the username input by the user is matched with the dictionary
            print("Sorry, that username does not exist. Please try again.")
            continue

        password = input("Enter your password: ")

        if password != users[username]: #check if the password input by the user is matched with the number in dictionary
            print("Incorrect password. Please try again.")
            continue

        print("Login successful!")
        return True
```



```python
def function():
  inventory = [] #dictionary that will store the items, unit price, and quantity

  while True: #while loop that will keep adding the input into the dictionary
    item_name = input("Enter item name: ")
    unit_price = float(input("Enter unit price: "))
    quantity = int(input("Enter quantity: "))
    inventory.append({"serial_number": len(inventory) + 1, "item_name": item_name, "unit_price": unit_price, "quantity": quantity

    add_more = input("Do you want to add more items? (y/n): ")
    if add_more.lower() != "y":
        break

  print("\nInventory List:\n")
  print("Serial Number\tItem Name\tUnit Price\tQuantity")# show the inventory list
  for item in inventory:
    print(f"{item['serial_number']}\t\t{item['item_name']}\t\t{item['unit_price']}\t\t{item['quantity']}")



  total_price = 0
  receipt_items = []

  while True:
    item_serial = int(input("Enter the serial number of the item you want to purchase: "))
    item = inventory[item_serial - 1]
    quantity = int(input("Enter quantity: "))

    if quantity > item['quantity']:
        print(f"Error: Only {item['quantity']} {item['item_name']} are available.")
        continue

    item_total_price = quantity * item['unit_price']
    total_price += item_total_price
    item['quantity'] -= quantity

    receipt_items.append({"item_name": item['item_name'], "unit_price": item['unit_price'], "quantity": quantity, "total_price":

    more_items = input("Do you want to purchase more items? (y/n): ")
    if more_items.lower() != "y":
        break

  print(f"\nTotal price: {total_price}")
  payment = float(input("Enter payment amount: "))
  change = payment - total_price
  print(f"Change: {change}")
```



```
  # Print receipt
  print("\nReceipt:")
  print("Item Name\tUnit Price\tQuantity\tTotal Price")
  for item in receipt_items:
    print(f"{item['item_name']}\t\t{item['unit_price']}\t\t{item['quantity']}\t\t{item['total_price']}")
  print(f"\nTotal Price:\t{total_price}")
  print(f"Payment:\t{payment}")
  print(f"Change:\t\t{change}")

  # Print updated inventory list
  print("\nUpdated Inventory List:")
  print("Serial Number\tItem Name\tUnit Price\tQuantity")
  for item in inventory:
    print(f"{item['serial_number']}\t\t{item['item_name']}\t\t{item['unit_price']}\t\t{item['quantity']}")




def main():
    while True:
        print("Welcome to the program!")
        print("Please choose an option:")
        print("1. Sign up")
        print("2. Log in")
        print("3. Quit")

        choice = input("Enter your choice (1-3): ")

        if choice == "1":
            sign_up()
        elif choice == "2":
            if log_in():
                function()
                break  # End the program after the inventory system is done
        elif choice == "3":
            break
        else:
            print("Invalid choice. Please try again.")


if __name__ == "__main__":
    main()
```

C. Work Breakdown

Student Name	**Task Assigned**	**Percentage of the Work Contribution**
Bandigan, Dustin Josh	**I. Introduction** **II. RRL** **III. Methodology** **VI. References**	50%
Yang, Yuankai	**Program Code** - Did all the coding from start to finish **Pseudocode** **Flowchart**	50%

D. Data Sheet

Student's Information Sheet

Subject: PROLOGI   Section: EQ3

Tri-Academic Year: 2022 – 2023   Professor: Mr. Ramon Ruiz

Personal:

Name: Dustin Josh Bandigan

Degree Program: Bachelor of Science in Computer Engineering

Address: Woodside Homes, New Manila, Quezon City

Cellphone Number: 09954862883

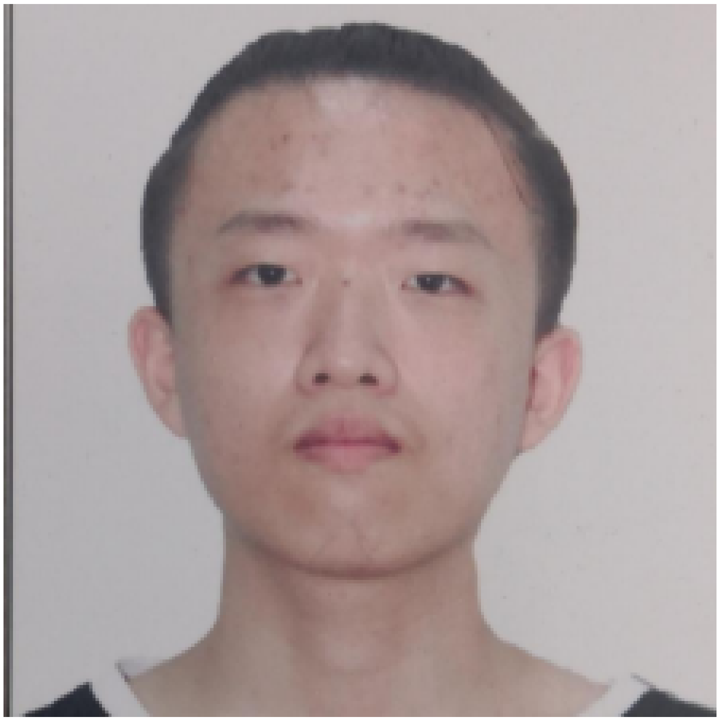
E-Mail: dustin_bandigan@dlsu.edu.ph

Birthday: June 3, 2003,   Age: 19

Family:

Father: Mark Benvo Bandigan   Occupation: Businessman

Mother: Katrina Bandigan   Occupation: Businesswoman



Name: Yuankai Yang

Course: CPE

ID: 12298596