**Rubric for Grading the Project (Refer to the Syllabus)**

| CRITERIA | EXEMPLARY (90-100) | SATISFACTORY (80-89) | DEVELOPING (70-79) | BEGINNING  (below 70) | WEIGHT |
| --- | --- | --- | --- | --- | --- |
| Experiment al Plan (Flowchart/ Algorithm)  (SO-PI: B1) | Experimental plan has supporting details and diagram/algorit hm that is stated and well explained | Experimental plan has supporting details and diagram/algorith m that is stated but not explained | Experimental plan is vague or brief. It has supporting details and does not have diagram/ algorithm | No experimental plan presented | 20% |
| Codes/Data/ Program | Data is well utilized in the program.  Program code are easy to read. Program output has no error.  Questions are answered completely and correctly | Data is somewhat utilized in the program.  Program code are easy to read. Program output has an output but logically incorrect. Some questions are answered completely and correctly | Data is not utilized in the program. It has a missing significant code/syntax in the program | No program presented | 30% |
| Use of Appropriate Tools and Techniques  (SO-PI: K1) | Appropriate tools and techniques are properly used for all aspects of the project | Appropriate  tools and techniques are used in most of the aspects of the project and all  of these are used properly | Appropriate  tools and techniques are used  in the majority of the aspects of the project  and all of these are  used properly | Appropriate  tools and techniques are used in less than half of the aspects of the  project and/or tools  are not used properly in at least  half the aspects of the project | 10% |
| Project  Documentation | Project documentation is orderly presented starting from statement of the problem, to objective of the project, followed by review of literature, design consideration, presentation of data or output and conclusion. The report was grammatically correct, logically presented and used the required format. | Project documentation is complete with statement of the problem, objectives, design consideration, presentation of data and output and conclusion.  The report had minimal grammatical errors and somewhat presented logically. The required format was used. | Project documentation is basically limited to algorithm presentation of data and output but no basis of the design was presented. The report had a lot of grammatical errors and not logically presented; the required format was barely used. | Project documentatio n is not reflective of algorithm design and/or characterizati on. The report had a lot of grammatical errors, was not logically presented and the required format was not used. | 30% |
| Project Presentation SO-PI: G2 | Project presentation is complete and backed up by complete design consideration, logic formulation and review of related literature | Project presentation is complete with algorithm simulation results backed up by design considerations. | Project presentation shows a system completely simulated but is not backed up by clear explanation of how algorithm was derived | Project presentation lacks clarity in terms of presenting and characterizin g the behavior of the algorithm | 10% |
| Total | | | | | 100% |

**DE LA SALLE UNIVERSITY - MANILA**

**Sale System**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

A Term Project

Presented to Mr. Ramon Stephen Ruiz

In Partial Fulfillment of the

Requirements for the Course Programming Logic And Design Laboratory

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

by

Enriquez, John Paolo

Evalla, Christian Matthew G.

Fernandez, Edge Matthew B.

EQ1

TF 7:30 - 10:30

April 15, 2023

**Table of Contents**

[**I. Introduction 6**](#_heading=h.lpfuamb30how)

[A. Background of the Study 7](#_heading=h.g8hq8s673cpa)

[B. Problem Statement 8](#_heading=h.gbv747nrrm3b)

[C. Objectives 8](#_heading=h.d3fog2iq7wah)

[**II. Review of Related Literature 9**](#_heading=h.62v2qtlpe2d)

[**III. Methodology 11**](#_heading=h.v7g2il1eqnlg)

[A. Conceptual Framework – IPO Chart (Input-Process-Output-Chart) 12](#_heading=)

[B. Hierarchy Chart 16](#_heading=h.5ggsspza26x1)

[C. Flowchart 17](#_heading=)

[D. Pseudocode 17](#_heading=h.89fbqj9fkift)

[**IV. Results 19**](#_heading=h.8f37tluiahuy)

[**V. Discussion of Results 36**](#_heading=h.u9i5l58ghft8)

[**Appendices 40**](#_heading=h.qchvvr7ebsmg)

[A. User’s Manual 40](#_heading=h.4nrt90dpbtzx)

[B. Source Code 41](#_heading=h.547nenoqay98)

[C. Work Breakdown 69](#_heading=h.u6ri07fzilw6)

[D. Personal Data Sheet 70](#_heading=h.k5hqsso09x1d)

**List of Figures**

[Figure 1, IPO for Enhanced security and anti-fraud protection. 13](#_heading=h.4ica8924n2gb)

[Figure 2, IPO for Menu Display 13](#_heading=h.x7cgegryj3fb)

[Figure 3, IPO for Action on orders 14](#_heading=h.5p8uejthipmi)

[Figure 4, IPO for Receipt 15](#_heading=h.lrnx7jrx1lzf)

[Figure 5, IPO for Save and load Cart 15](#_heading=h.1tnqpyisdcak)

[Figure 6, Hierarchy Chart 16](#_heading=h.pg5jj7d2gkze)

[Figure 7, Flowchart of code 17](#_heading=h.pf6om1996qjs)

[Figure 8.1, Set 1 - Account Login 20](#_heading=h.uip8hbuu39hr)

[Figure 8.2, Set 1 - Main Menu 21](#_heading=h.1t3hr03zqqi)

[Figure 8.3, Set 1 - Items Menu 21](#_heading=h.vtertgetbus8)

[Figure 8.4.a, Set 1 - Adding to cart 21](#_heading=h.jixfxv4a6ud5)

[Figure 8.4.b, Set 1 - Adding to cart 22](#_heading=h.jh41gq6vfbr6)

[Figure 8.5, Set 1 - Checkout 22](#_heading=h.t5sb7c6pgcfy)

[Figure 8.6, Set 1 - Payment Method 23](#_heading=h.offlg0uhm2xe)

[Figure 8.7, Set 1 - Placed an order 24](#_heading=h.gcmbar2gqvqn)

[Figure 8.8.a, Set 1 - Cart Saving System Test 25](#_heading=h.ynxpehwjekqn)

[Figure 8.8.b, Set 1 - Cart Saving System Test 25](#_heading=h.5zuhpkwfq63r)

[Figure 8.9, Set 1 - Transaction History Test 26](#_heading=h.dqxupkyihdpp)

[Figure 9.1, Set 2 - Account Login 27](#_heading=h.7hydgtgt92f7)

[Figure 9.2, Set 2 - Main Menu 27](#_heading=h.yei2sti3lkfc)

[Figure 9.3, Set 2 - Items Menu 28](#_heading=h.jl8rnq7du6lg)

[Figure 9.4, Set 2 - Adding to cart 28](#_heading=h.t8l20yrfrgov)

[Figure 9.5, Set 2 - Clearing cart 28](#_heading=h.pxqw9w8h818n)

[Figure 9.6, Set 2 - Back to Main Menu 29](#_heading=h.f21lhbvv28t1)

[Figure 9.7, Set 2 - Back to Items Menu 30](#_heading=h.yrq2n6fdjj2w)

[Figure 9.8, Set 2 - Adding to cart 30](#_heading=h.8tkt3gxeofpv)

[Figure 9.9, Set 2 - Checkout and Payment Method 31](#_heading=h.3c1dueq6gii9)

[Figure 9.10, Set 2 - Information Displayed 31](#_heading=h.tg8dvfar9nzc)

[Figure 9.11, Set 2 - Logging Out 31](#_heading=h.tm7favwyic4p)

[Figure 10.1, Set 3 - Signing In 32](#_heading=h.t5rl5nngcj0a)

[Figure 10.2, Set 3 - Main Menu 33](#_heading=h.rvw3i1a43901)

[Figure 10.3, Set 3 - Items Menu 33](#_heading=h.b6o484f3d2nw)

[Figure 10.4.a, Set 3 - Adding to cart 33](#_heading=h.upcawm66iaj7)

[Figure 10.4.b, Set 3 - Adding to Cart 34](#_heading=h.aqgepmfzsl1s)

[Figure 10.5, Set 3 - Checkout 34](#_heading=h.r5c10q9lgp6n)

[Figure 10.6, Set 3 - Account Details 35](#_heading=h.runz1ktjvzum)

[Figure 10.7, Set 3 - Information Displayed 36](#_heading=h.iybq6ai94hde)

# Introduction

The proponents of this project will create a POS Software that is used by the customers to buy products from a Computer Hardware store. The software can be used for online stores. The POS software will be capable of adding customer orders, and accepting payment. The POS Software will make it easier to see previous transactions as it can save transactions of different customers for the manager, or even the customers themselves can see their previous transactions in their accounts. The concepts of Arrays and Modules, which are fundamental topics covered in the PROLOGI syllabus, will be utilized in this project. These concepts serve as the basic building blocks of the project, and their successful integration will enable the project to achieve its objectives with ease. By leveraging these concepts, we aim to create a robust and efficient solution that effectively addresses the problem at hand.

## Background of the Study

Cash Registers are commonly used by businesses to compute tax, tips, and discounts. The features of cash registers can vary depending on the price range but oftentimes they are still cheaper than any POS system which makes them good for small businesses. Cash registers are cheap, easy to use, safe, and secure (Dublino, 2023). A cash register would be enough for small businesses but if you want to be able to cater to and attract more customers, There is an alternative now that is called the Point of Sale (POS). A point of sale can be a device or a place where a customer can easily select and pay for goods that are sold by a business. Sale taxes are also payable here and transactions are much more convenient and easier for employees to manage (Hayes, 2022). POS systems can be implemented in both physical and online stores and purchase histories, and receipts can be easily saved in local or cloud servers, unlike cash registers which can lose data (Dublino, 2023).

According to Glend (2022), 5 things can happen if you don’t use a POS System in a restaurant. Despite it being used for restaurants this is still applicable even for a business like this which is a computer hardware store. First is an outdated payment infrastructure. An outdated payment option can provide a worse and more hassling experience to customers for example who want to pay with payment options like GCash. Second, it can complicate financial reports, due to the manual counting of old receipts. Third is that it is more difficult to integrate with an inventory system or any other systems used in businesses. POS Systems are easier to integrate with inventory systems due to them being easily connected to different systems unlike traditional cash registers they generally don’t have the technology to be linked with other business systems sometimes even connecting to the internet is difficult. Fourth is that service to customers will be a lot slower for both the employees and customers. Lastly, another thing that can happen is that receipts can be incomplete for cash registers. It's possible that the discount on products wouldn’t be counted in the receipt even though you paid less than the actual amount. It makes it difficult to track financial transactions for customers and employees.

## Problem Statement

In today's fast-paced world, technology advances rapidly, and customer expectations shift quickly. As a result, it is essential to keep up with these changes and offer convenient and efficient shopping experiences. Cash registers provide to be insufficient for dealing with the amount of customers and of course for situations like online shopping, it is not used at all. a better alternative is needed. The computer hardware industry needs a better POS system that will help adjust to the new ways of shopping in the industry. Otherwise, it will be difficult for managers to gain insights into sales trends and for customers to buy items from their shops.

## Objectives

**C.1 General Objectives**

1. To improve our skills in creating flowcharts and pseudocode.
2. To enhance our proficiency in the Python programming language.
3. Apply all the relevant concepts and topics covered in the PROLOGI syllabus to the program.

**C.2 Specific Objectives**

1. Develop a Point of Sale (POS) program for a computer hardware shop.
2. Point of sale should have the following systems:
   1. Enhanced security and anti-fraud system
   2. Menu system
   3. Cart System (with and save and load function)
   4. Receipt System (previous transactions should be viewable)

**D. Significance of the Study**

This study is conducted to benefit the following:

Students. This research could be utilized by students undertaking similar studies as a reference and guide.

Customers. Customers can benefit from this by having access to more efficient and user-friendly POS software that enhances their shopping experiences when purchasing computer hardware items.

Store Managers. Store managers can benefit from an advanced and efficient POS system that allows them to analyze sales trends accurately, gain insights into consumer behavior, and improve overall business management.

Computer Hardware Stores. Computer hardware stores can benefit from a highly advanced and effective POS system that improves their ability to provide a better shopping experience to clients and track sales trends accurately.

Other Stores. This research can inspire other stores to adopt an advanced and efficient POS system, which can improve the shopping experience for customers and help managers track sales trends more accurately. By implementing such a system, stores can make informed business decisions.

Future Researchers. The findings of this study can be valuable to future researchers, and may serve as a foundation for developing new theories in the field of learning.

# II. Review of Related Literature

A Point of Sale (POS) software, in accordance with Chavan & Waghmare (2020), offers business owners a practical way to track sales and identify consumers. Every time an order is fulfilled, the software automatically updates its correct inventory record. Additionally, it can track clients, handle credit card payments, print receipts, and carry out other tasks. POS software improves operations and enables owners to make more informed business decisions by easing the checkout process and recording crucial data.

According to Squareup (2023), A POS system typically includes features like processing transactions, tracking sales data, managing, and reporting inventory. A point-of-sale (POS) system is a hardware and software combination that enables companies to handle revenues, stock, and customer information. Using a POS system increases efficiency, accuracy, and customer satisfaction. Squareup advises businesses to select the right POS system according to their size, market, and specific needs. Overall, a POS system can be a valuable asset to companies looking to simplify their operations and improve their customer service.

The StoreHub (2021) article, "Top 10 Benefits of a POS System for Your Retail Business", discusses the advantages of having a point-of-sale (POS) system in a retail business. According to the article, a POS system can assist a business in streamlining operations, increasing efficiency, reducing errors, improving inventory management, and providing valuable data insights. A POS system can also assist businesses in saving time and money, providing better customer service, and increasing sales through cross-selling and up - selling. The importance of a POS system for any retail business seeking to grow and succeed in today's competitive market is emphasized in the article's conclusion.

A study in Bangladesh (2011), studied the implementation of Web-Based POS systems for small and medium enterprises in Bangladesh. Small and medium enterprises in their country were the main driving force for the economy. It reduces poverty and creates employment opportunities so implementing a POS system could enhance the business opportunities for their country. The study found that implementation of the web-based POS system for small and medium enterprises will greatly assist them and that there are many opportunities to introduce this kind of technology.

According to Magaziner (2022), Restaurant services were greatly affected by the pandemic. Restaurant owners were pondering what to do to survive and had to think about how to get customers while keeping their customers safe. The solution they thought of was in the POS system. Reworking and reconfiguring the POS systems of the restaurant was essential for the restaurant to survive. They adjusted the system for things like accepting tips, providing discounts, and flexible order options. What can be gathered from this research is that the POS System by itself is not good enough. It can be changed according to the user’s needs and improved to cater to services properly.

According to Roding (2023), Retailers are investing in in-store technology to enhance the shopping experience and increase customer happiness, customer loyalty, and shopping frequency. In the first quarter of 2021, investments in retail technology increased by a factor of three to $25 billion, allowing employees to spend more time on higher-impact tasks. However, the subject of POS technology's tangible application-related added value or consumer perception of different POS technologies in the salesroom remains a major issue for both producers and retailers.

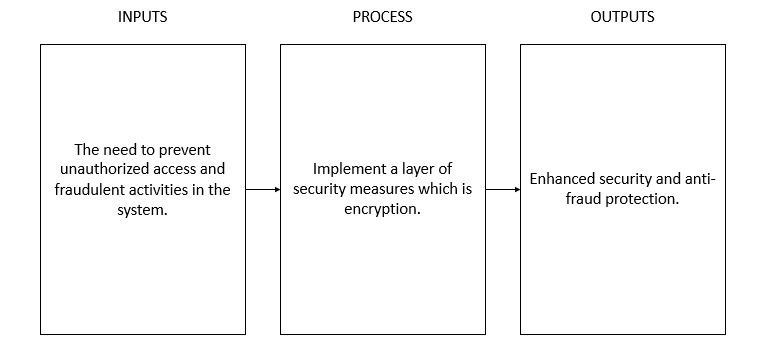
According to Smith, John(2022). The benefits of adopting a POS system in retail operations are covered in "How Point of Sale (POS) Systems Enhance Retail Operations". It demonstrates how a POS system may better inventory management, boost revenue, increase accuracy, shorten checkout times, and enhance customer service. The article gives a thorough summary of the advantages of utilizing a POS system, making it a useful resource for information for companies thinking about making this technology investment. This article is a helpful resource for anyone interested in learning about the significance of a POS system in retail operations because it has a reliable source and a full description of the benefits

# III. Methodology

The creation of the POS software project is going to stick to a thorough approach that covers all of the crucial elements required for effective software development. To make sure the software satisfies the needs and expectations of the stakeholders, including customers, management, and staff, the initial phase entails gathering requirements from them. The design phase is the following stage, when the requirements acquired will be used to construct the software architecture, user interface, and database structure. The writing of the POS software's code will take place during the development phase using the chosen programming language(s) and software development tools. The program will undergo thorough evaluation to guarantee that it satisfies the functional and non-functional criteria, and any problems will be quickly fixed. The program will be installed in the target environment, whether it is an online store or a physical store, when testing is over. To keep the software current and functional, on-going maintenance and support will be offered. The fundamental software building blocks of Arrays and Modules will be used throughout the project to make sure it is effective, scalable, and maintainable. With the help of a practical and user-friendly point-of-sale system that facilitates transactions and simplifies sales tracking, this methodology seeks to develop a reliable and effective solution to the issue at hand.

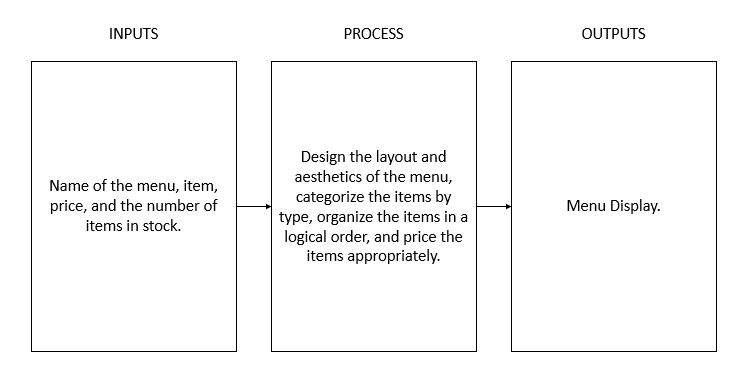
## Conceptual Framework – IPO Chart (Input-Process-Output-Chart)

1. Enhanced security and anti-fraud protection.



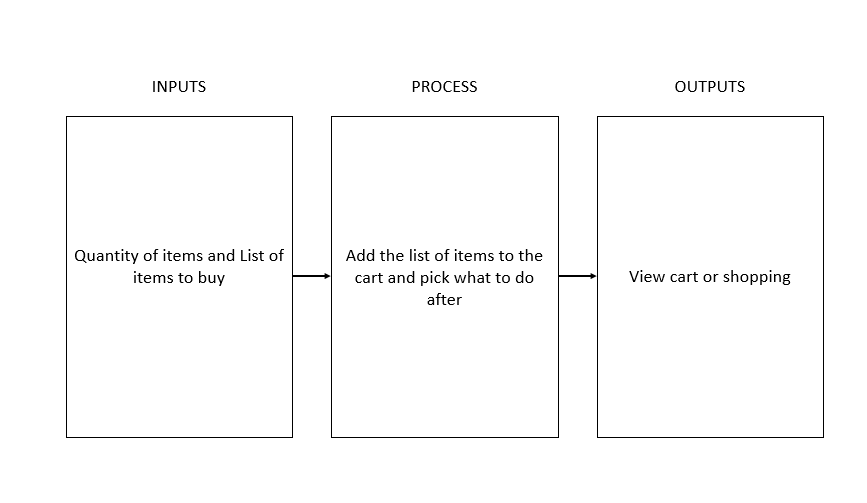
### *Figure 1, IPO for Enhanced security and anti-fraud protection.*

1. Menu Display



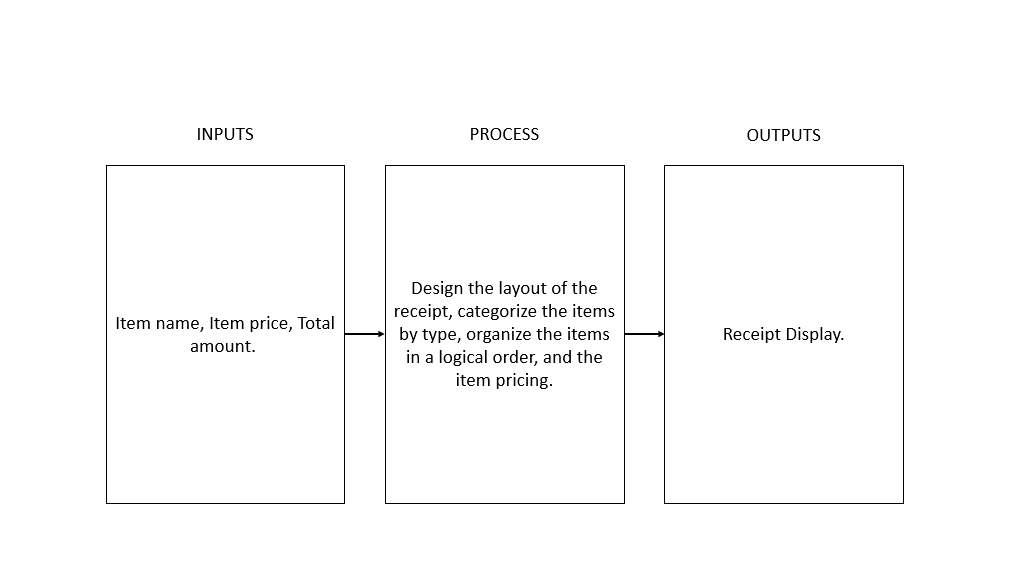
### *Figure 2, IPO for Menu Display*

1. Action on Orders



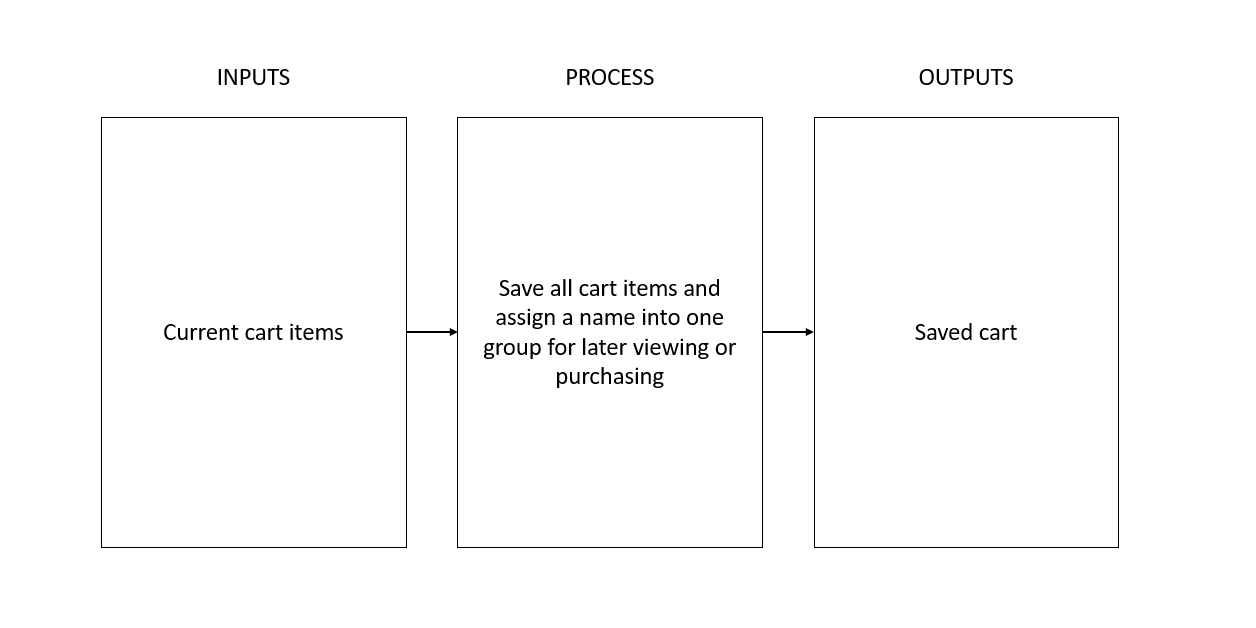
### *Figure 3, IPO for Action on orders*

1. Receipt

****

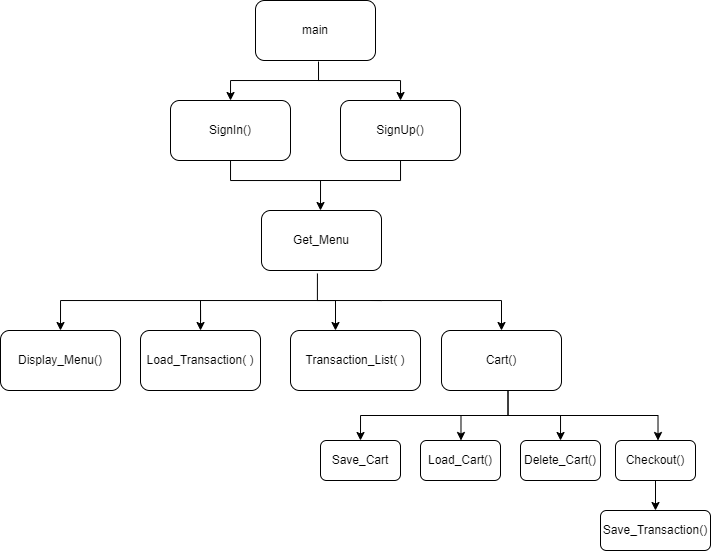
### *Figure 4, IPO for Receipt*

1. Others
2. Save and load cart with name system



### *Figure 5, IPO for Save and load Cart*

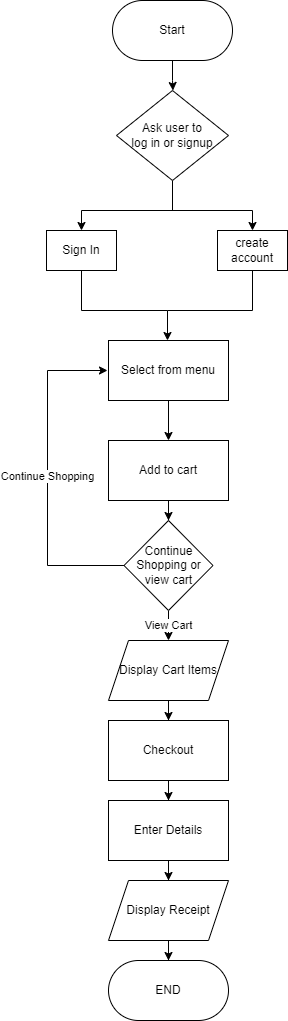
## B. Hierarchy Chart



### *Figure 6, Hierarchy Chart*

The main code runs all of the modules. SignIn() contains the code for signing in an account, SignUp() contains the code for registering an account. Get\_Menu, gets the items from the menu and puts it into an array for the item name, price, and description. Display\_Menu(), displays the menu. Load\_Transaction, loads the receipt of a previous transaction of the current account. Transaction\_List() displays the history of transactions of the current account. Cart(), displays the cart and returns the array of the cart as well. Under Cart(), you can Save\_Cart(), which from its names saves it with a name of your choice. Load\_Cart(), allows you to load the carts you have saved. Delete\_Cart(), allows you to delete the cart you have previously saved. Finally, you can checkout using Checkout() and the transaction is automatically saved using Save\_Transaction().

## C. Flowchart

****

### *Figure 7, Flowchart of code*

## D. Pseudocode

// Initialize variables

menu = {} // Dictionary to store menu items

cart = [] // List to store customer's items

total = 0 // Total amount of customer's purchase

is\_fraudulent = False // Flag to indicate if transaction is suspicious

// Populate menu

menu["TravelPro™ G15 12G (GTX 1650)"] = 52800.00

menu["Xitrix GX3 (Z790) Elite Gaming PC"] = 93800.00

menu["Xitrix® GX1R NANO AMD Gaming PC"] = 46800.00

menu["TravelPro™ U1412 Premium Ultrabook"] = 42800.00

menu["WFP-2413 24" Full HD 75Hz Professional VA Monitor"] =6800.00

menu["DeskFrame™ E305R AMD Desktop"] = 20800.00

menu["Xitrix® GX3R (X670) Gaming PC"] = 116800.00

menu["Xitrix® GL1 (H610) Gaming PC"] = 24800.00

menu["Xitrix® GX1 NANO Gaming PC"] = 46800.00

menu["Xitrix® G27 12G Gaming AIO PC"] = 39800.00

menu["DeskFrame™ E310 12G CFF"] = 24800.00

menu["Xitrix® GX2 (Z690 D4) Gaming PC"] = 67800.00

menu["TravelPro™ GX15 (RTX3070) 300Hz"] = 102800.00

menu["TravelPro™ M15"] = 46990.00

menu["CANON G3020 Wireless 3IN1(Print, Copy, Scan) CIS Printer"] = 11400.00

menu["Xitrix® GX48 48" 4K UHD 138Hz OLED Gaming Monitor"] = 79800.00

menu["Xitrix® GX2R Creator PC"] = 246800.00

menu["OptoFrame™ W130 SFF"] = 55000.00

menu["Xitrix® 90W LAPTOP Power Adapter"] = 1800.00

menu["Crucial P5 Plus PCIe 4.0 M.2 2280SS Gaming SSD"] = 4800.00

// Display menu

print("Welcome to our restaurant!")

print("Please choose from the following menu items:")

for item in menu:

print(item + " - $" + str(menu[item]))

// Customer adds items to cart

while True:

item = input("Enter item name (or 'done' to finish): ")

if item == "done":

break

elif item in menu:

cart.append(item)

total += menu[item]

print(item + " added to cart.")

else:

print("Sorry, we don't have that item on the menu.")

// Print total and prompt for payment method

print("Your total is $" + str(total))

payment\_method = input("Enter payment method (cash or card): ")

// Process payment

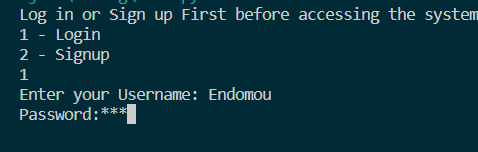
print("Thank you for your purchase!")

else:

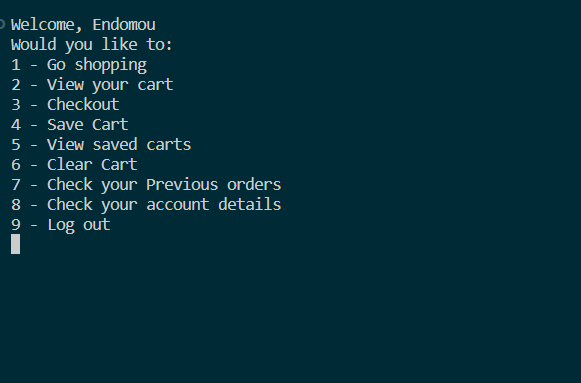
print("Sorry, we don't accept that payment method.")

# IV. Results

1. Set 1

****

### *Figure 8.1, Set 1 - Account Login*

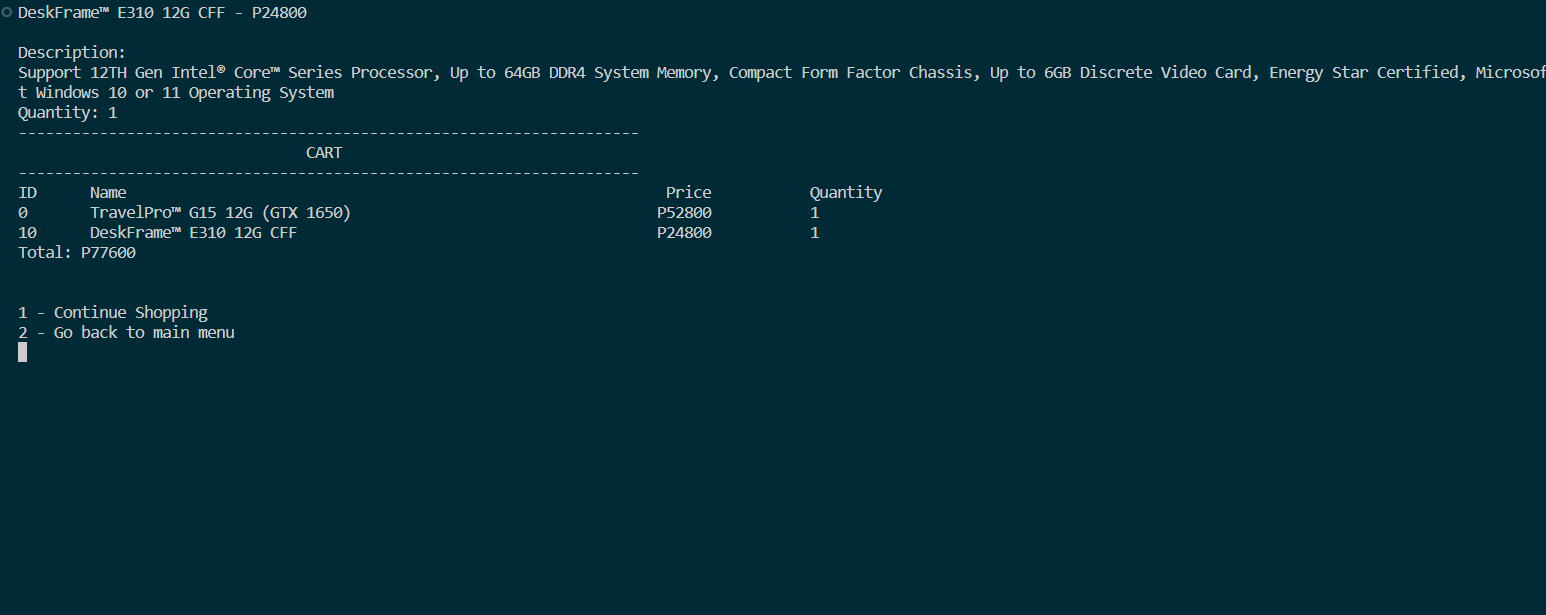
**

### *Figure 8.2, Set 1 - Main Menu*

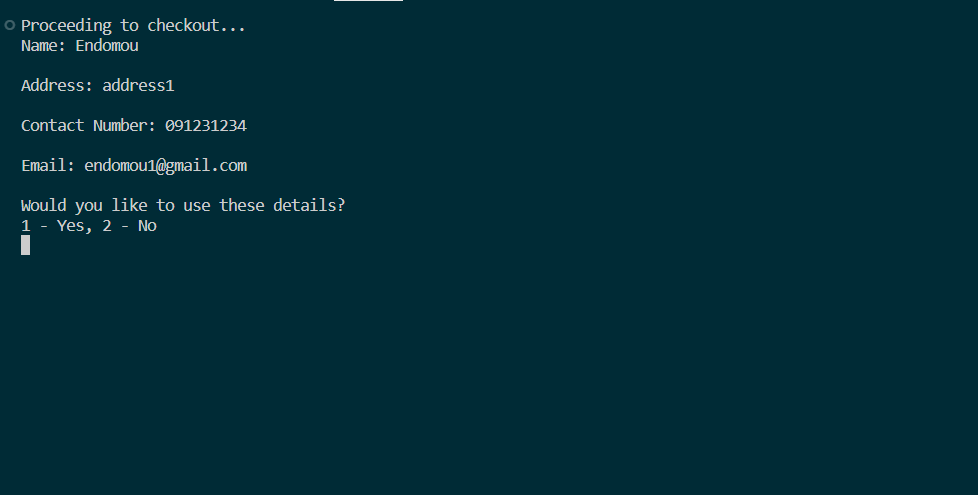
### *Figure 8.3, Set 1 - Items Menu*

### 

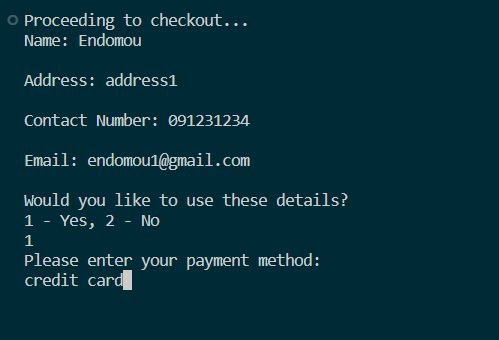
### *Figure 8.4.a, Set 1 - Adding to cart*

****

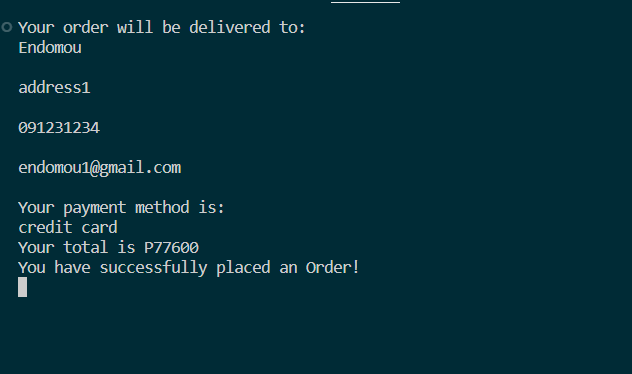
### *Figure 8.4.b, Set 1 - Adding to cart*

****

### *Figure 8.5, Set 1 - Checkout*

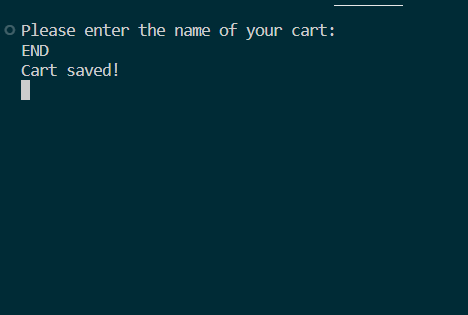
****

### *Figure 8.6, Set 1 - Payment Method*

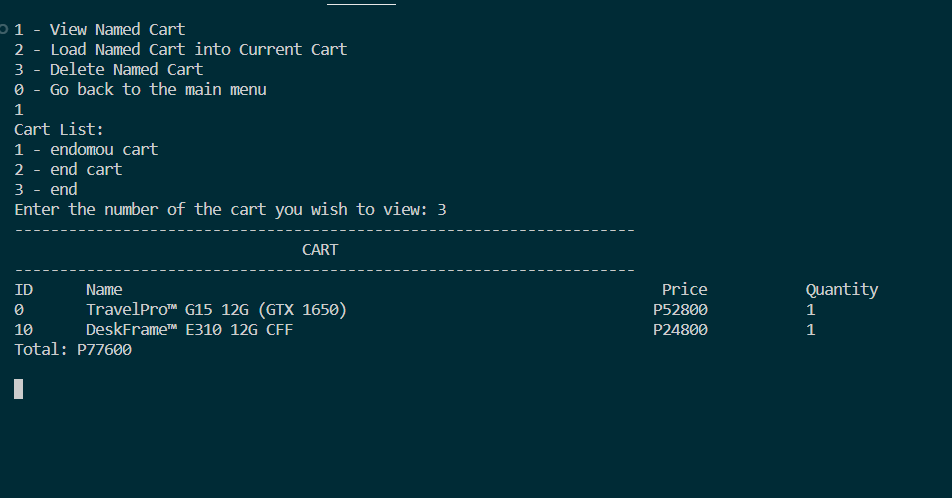
****

### *Figure 8.7, Set 1 - Placed an order*

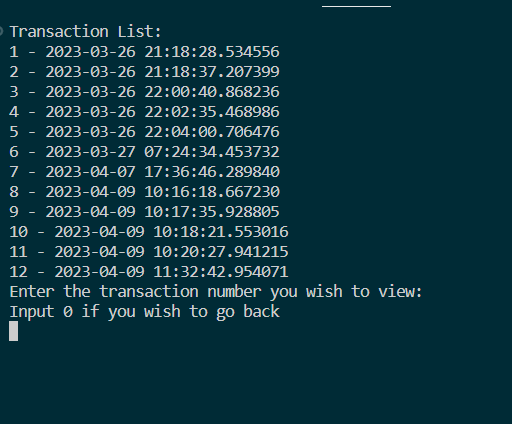
Cart Saving System Test

****

### *Figure 8.8.a, Set 1 - Cart Saving System Test*

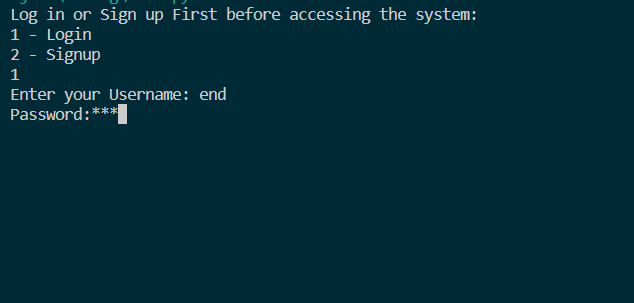
****

### *Figure 8.8.b, Set 1 - Cart Saving System Test*

****

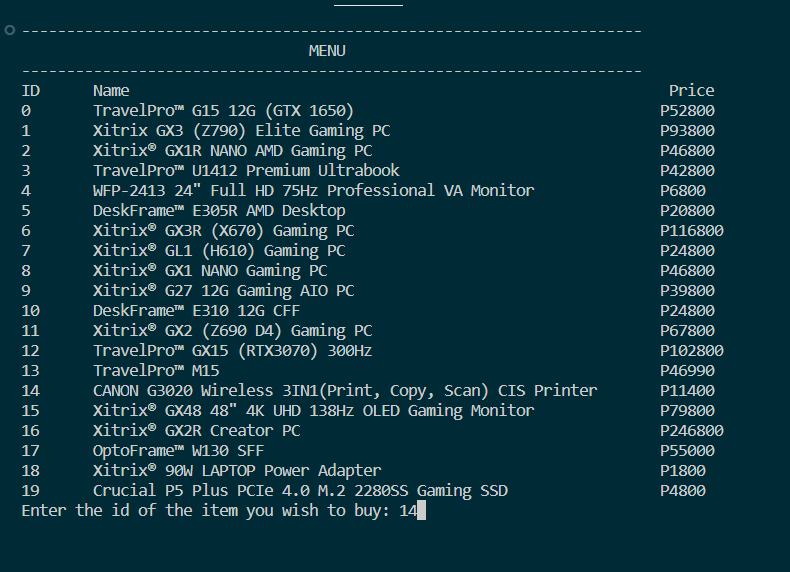
### *Figure 8.9, Set 1 - Transaction History Test*

1. Set 2:

****

### *Figure 9.1, Set 2 - Account Login*

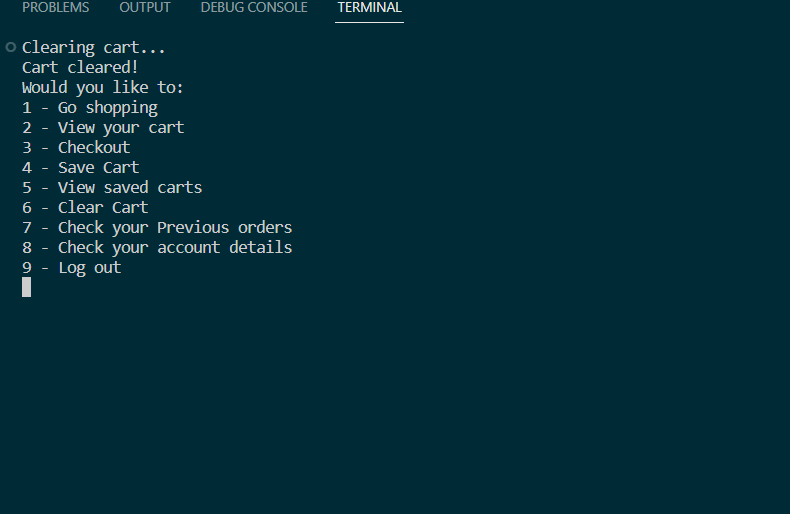
### *Figure 9.2, Set 2 - Main Menu*

****

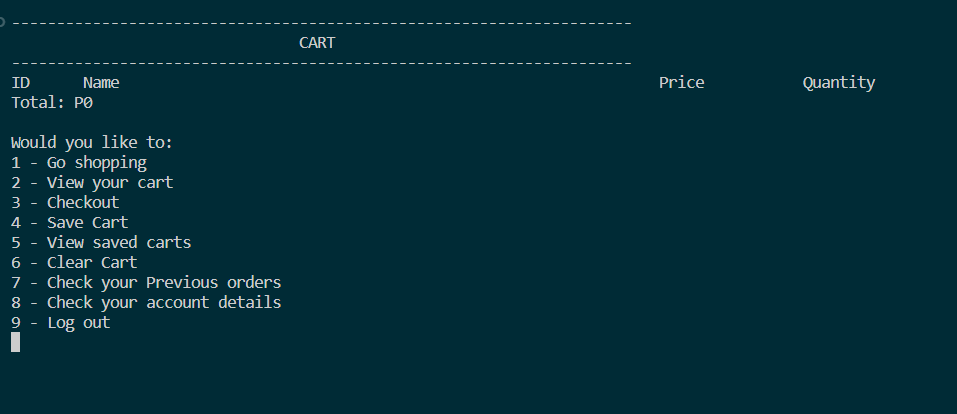
### *Figure 9.3, Set 2 - Items Menu*

****

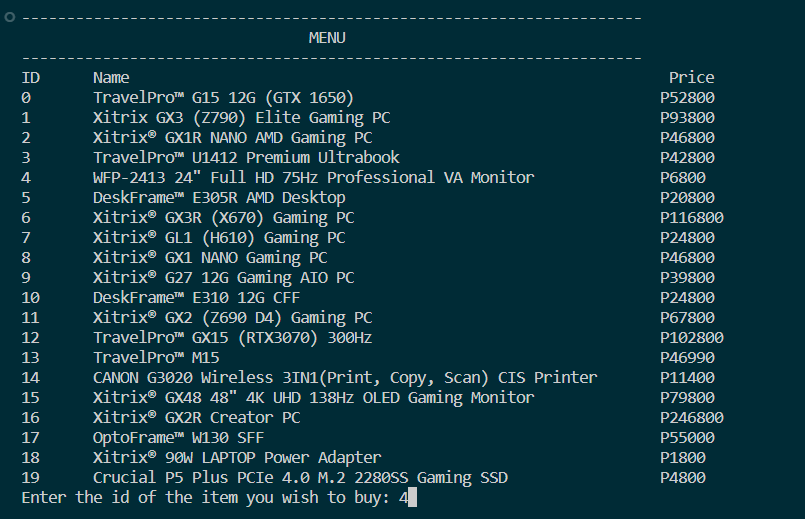
### *Figure 9.4, Set 2 - Adding to cart*

****

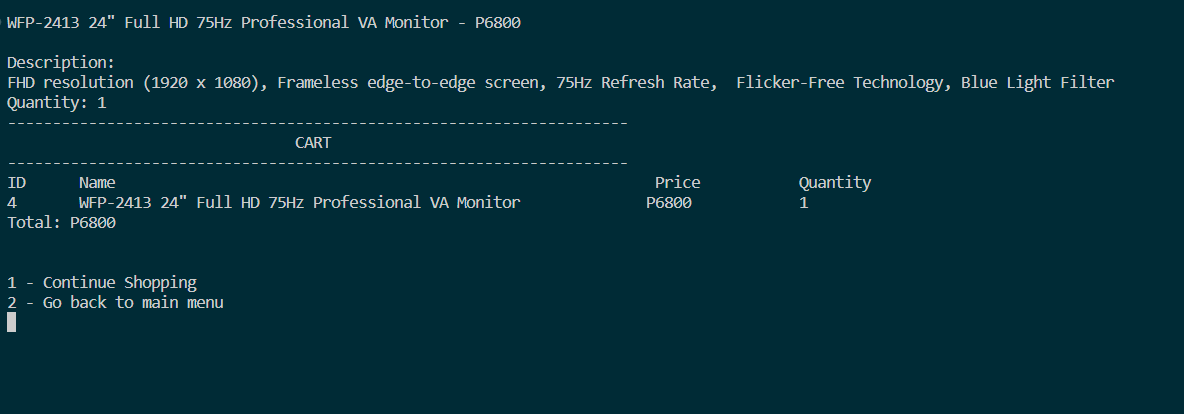
### *Figure 9.5, Set 2 - Clearing cart*

****

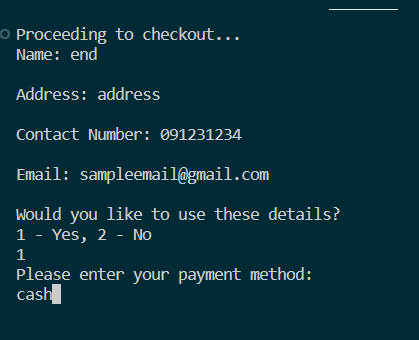
### *Figure 9.6, Set 2 - Back to Main Menu*

****

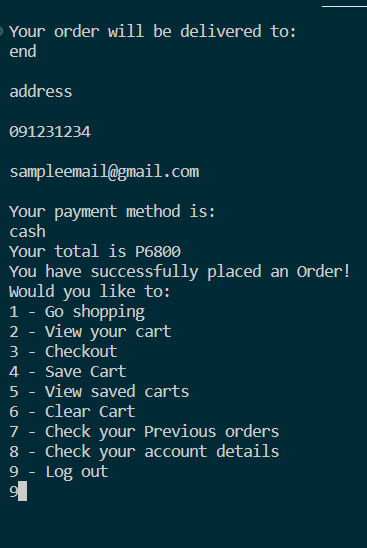
### *Figure 9.7, Set 2 - Back to Items Menu*

****

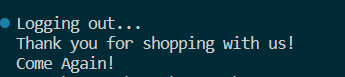
### *Figure 9.8, Set 2 - Adding to cart*

****

### *Figure 9.9, Set 2 - Checkout and Payment Method*

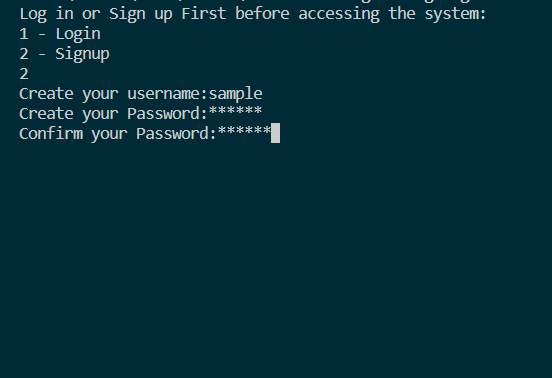
****

### *Figure 9.10, Set 2 - Information Displayed*

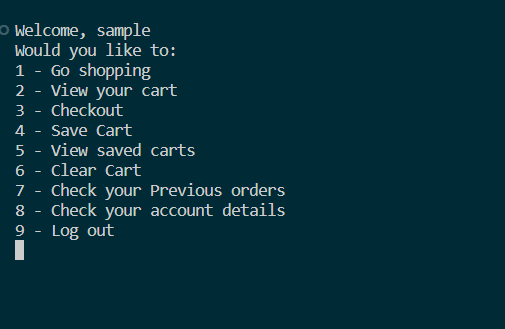
****

### *Figure 9.11, Set 2 - Logging Out*

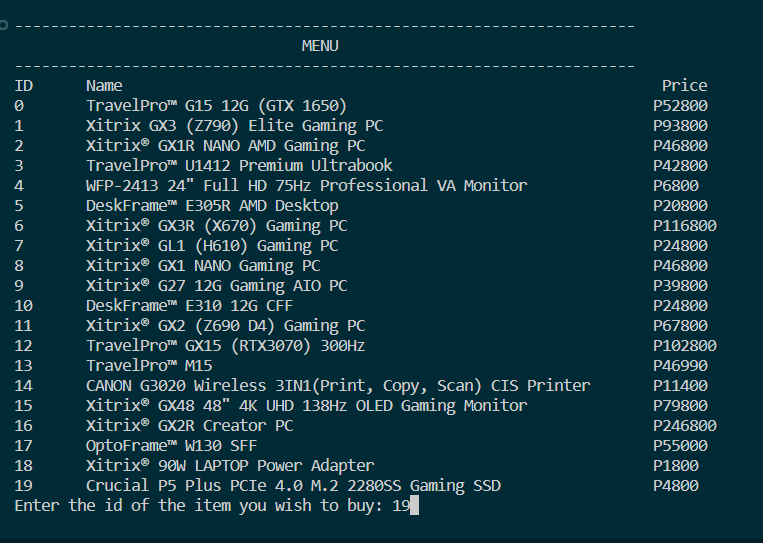
1. Set 3

****

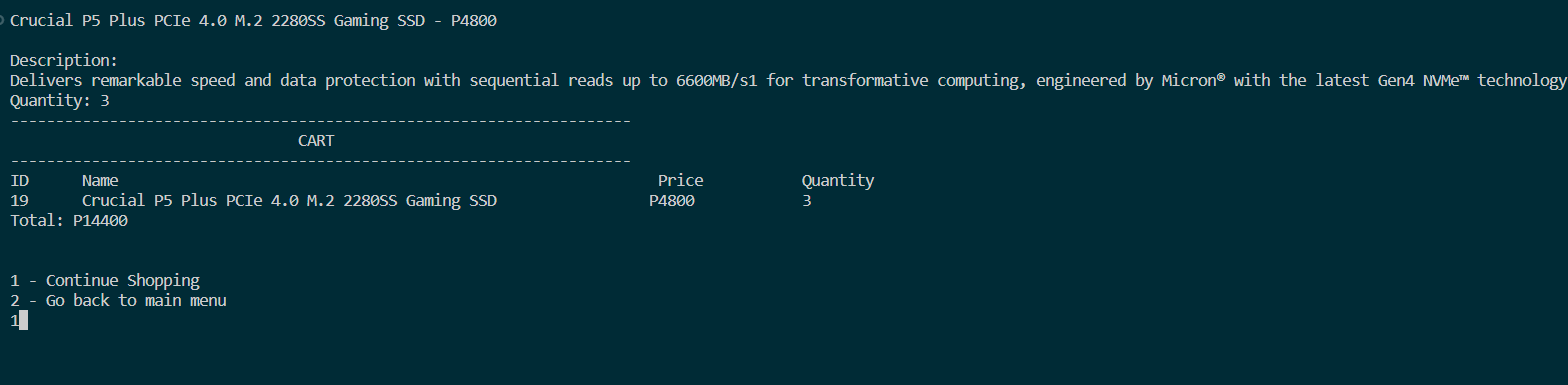
### *Figure 10.1, Set 3 - Signing In*

****

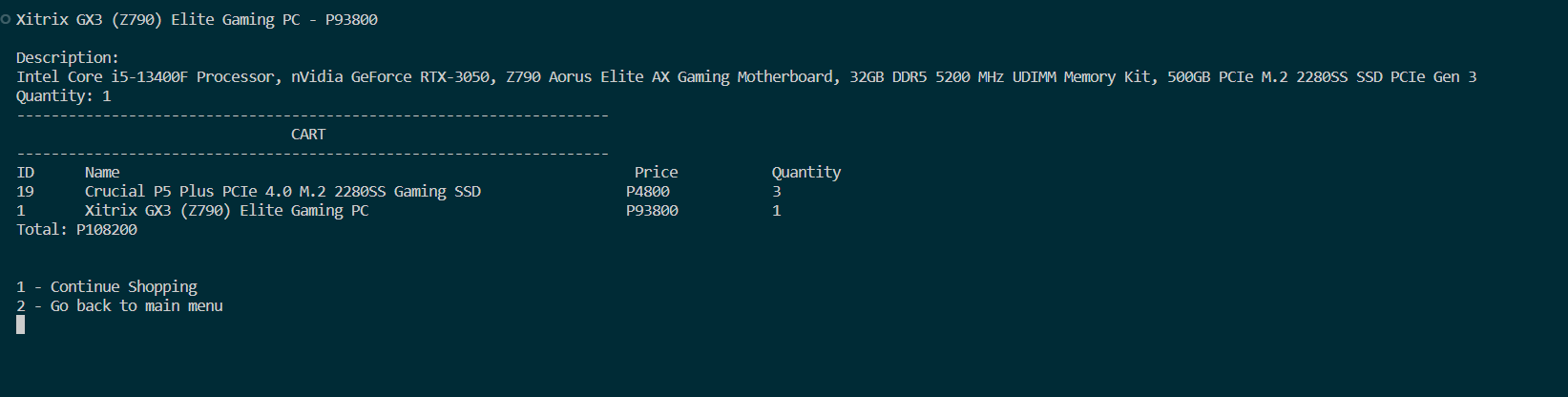
### *Figure 10.2, Set 3 - Main Menu*

****

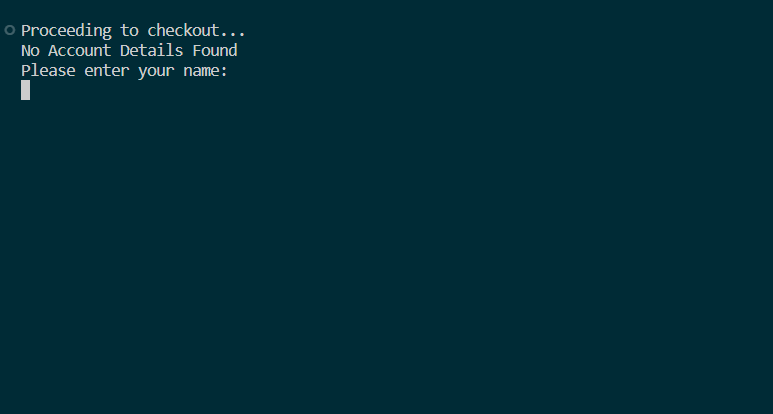
### *Figure 10.3, Set 3 - Items Menu*

****

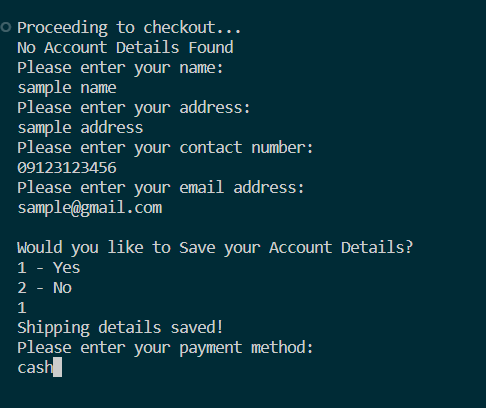
### *Figure 10.4.a, Set 3 - Adding to cart*

****

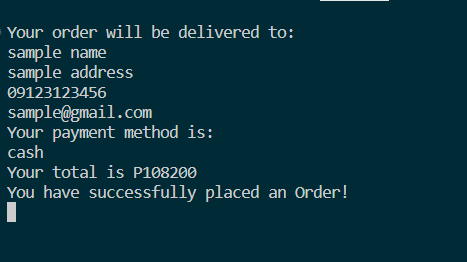
### *Figure 10.4.b, Set 3 - Adding to Cart*

****

### *Figure 10.5, Set 3 - Checkout*

****

### *Figure 10.6, Set 3 - Account Details*

****

### *Figure 10.7, Set 3 - Information Displayed*

# V. Discussion of Results

1. Set 1

You logged in to your account by entering your username and password. Once logged in, you were directed to the homepage where a menu displayed different options to choose from. You selected "Go Shopping" by inputting the corresponding number 1. The shopping page displayed the menu with various items and you added some to your cart. You proceeded to the checkout where your provided personal information is shown and you were asked about your payment information. The system confirmed your information and displayed the order summary, including the total cost of the items you ordered. You successfully completed the purchase using a credit card. After completing the purchase, the system prompted you to name your cart, which you did. The cart was then saved to your list, allowing you to view it later if needed. You also explored the transaction history feature and were able to view all your previous purchases. The code then ends here.

1. Set 2

You began your online shopping experience by securely logging in to your account using your username and password. Once you were directed to the homepage, you were presented with a menu displaying various options. You selected the option to go shopping by entering the corresponding number 1. While browsing the available items, you added an item to your cart but then cleared it. This action returned you to the homepage where you again clicked the option to 1 - go shopping. This time, you added an item to your cart and proceeded to checkout. At checkout, the system displayed your provided personal information and you opted to pay in cash. The system confirmed your information and calculated the total cost of your purchase. You successfully completed your transaction and logged out of your account.

1. Set 3

You didn't have an account yet, so you signed up for one. After signing up, the system directed you to the homepage where you were presented with a menu of options to choose from. You selected "Go Shopping" by inputting the corresponding number 1. The shopping page displayed various items, and you added some to your cart. When you proceed to checkout, the system informs you that it didn't have your information on file yet since you had just created the account. It then prompted you to input your personal and payment information, which the system saved for future purchases. Once you confirmed your information, the system calculated your total, and you successfully placed an order using cash. The code ends here.

The main.py contains most of the code and there are 4 modules; cart.py, account.py, menu.py, receipts.py. The cart module includes all of the code regarding the cart like the save cart, load cart, delete cart, find cart, and the checkout as well. For the Cart it takes the arrays to display the cart and get the total. find cart is used to find and load the cart from the database. For most of the cart modules they use file handling and arrays. The Save Cart system works by saving the items of the cart to a txt file called “<username>\_cart.txt” where username would be based on what account is used. which would be then loaded for the load cart system For the account module, it uses file handling and saves the accounts to a txt file “accounts.txt”. The account module also makes use of a module of python called maskpass, where it allows the entered input to turn into any characters you want. This is used to mask the password of the user input. The txt file “accounts.txt”is also used to find registered accounts using a search algorithm. Account details are also saved in a file based on the username of user <username>\_account.txt. For the receipt system. A similar search algorithm that was used in accounts is used to find the dates of the file where it checks the whole file and checks for dates of transactions. File handling again is used to save previous transactions within the POS system. The transactions for each user account are saved in different files based on the username of the user. It is saved in a file named   
<username>\_receipts.txt. Lastly, the menu makes use of a module called openpyxl it allows the program to read from an excel file to get the items of the menu to make the code easily scaleable in cases the menu is changed, All of these modules are now called in the main.py and this contains the main menu of the shopping system.

**VI. Analysis, Conclusion, and Future Directives**

**Analysis**

A comprehensive system that covers numerous issues facing the sector, point of sale (POS) software was created for a computer hardware store. The program features a user-friendly menu system, a cart system with a save and load feature, and a receipt system that enables users to check their previous transactions in addition to improved security and anti-fraud measures. The software uses foundational ideas and concepts from the PROLOGI curriculum, like arrays and modules, to produce a reliable and effective solution. Through a simulated online shopping experience that includes login into the account, perusing items, adding them to the basket, and finishing a transaction, the program's functionality was evaluated. All of these tasks were completed effectively by the program, giving clients a flawless shopping experience.

**Conclusion**

The point-of-sale software created for a computer hardware store is a vast improvement over the sector's current systems. Numerous issues are addressed by the program, including bad customer service, insufficient sales trend analysis, and insufficient security measures. The program is not only reliable and effective but also expandable and flexible because it makes use of the ideas and subjects covered in the PROLOGI syllabus.

**Recommendation**

The advice is to keep funding the system's extension and enhancement in light of the system's excellent performance and potential, which was created for the computer hardware store. The following areas should be researched in order to do this:

First and foremost, the system's integration with inventory management systems needs to be prioritized in order to support real-time stock monitoring and automatic database updates. This would greatly lower the likelihood of stock-outs or overstocking, resulting in the store's inventory levels being at their ideal levels.

Second, the program would gain significant insights into sales trends and patterns with the addition of real-time analytics capabilities, empowering shop managers to make wise choices regarding pricing, promotions, and inventory management. Store managers can proactively adjust their offers to match the unique demands of their clients and boost total sales with the help of this information.

Finally, in order to maximize the software's commercial potential, it should be tailored to the particular wants and demands of different computer hardware sellers. With this flexibility, businesses could modify the software to meet their own requirements, creating a point-of-sale solution that was more specialized and efficient.

In conclusion, firms in the computer hardware sector would greatly profit from investments made in the creation and growth of point-of-sale software for computer hardware stores. The system's adaptability, scalability, and flexibility make it a perfect starting point for point-of-sale technology research and development in the future, with potential applications in a variety of other industries, including healthcare, hospitality, and retail.

**Future Directives**

By adding new features and functionalities, the point-of-sale system created for the computer hardware store can be improved even more. The program, for instance, can be connected to inventory management systems to monitor stock levels and update the database when things are sold. Real-time analytics can also be added to the program to help store managers track sales trends and come to wise judgments. The software can also be further altered to accommodate the particular requirements of various computer hardware retailers.

On the foundation of this initiative, future researchers can create cutting-edge POS systems for several sectors, including hospitality, healthcare, and retail. The flexibility and adaptability of the program make it a perfect base for upcoming studies in this field.

**References**

Chavan, S. & Waghmare, N. (2020). A conceptual study of P.O.S. (Point of Sale) system, improving profitability of Food & Beverage Department in a Restaurant. International Journal of Creative Research Thoughts (IJCRT). Retrieved from <https://ijcrt.org/papers/IJCRT2006464.pdf>

Deutch, K. (2022). What is a POS System? Squareup. Retrieved from <https://squareup.com/us/en/townsquare/what-pos-system>

Dublino, J. (2023). Cash Register Buying Guide: POS vs. Cash Registers vs. Tablet mPOS <https://www.business.com/articles/cash-register-vs-pos-system/>

Glend, V. (2022). 5 things that will happen if you don’t use POS System in your restaurant. <https://www.hashmicro.com/blog/5-things-that-will-happen-if-you-dont-use-pos-system-in-your-restaurant/>

Hayes, A. (2022) What Is a POS System and How Does It Work? <https://www.investopedia.com/terms/p/point-of-sale.asp>

Kim, Y. & Lim, J. (2011). A POS system based on the remote client-server model in the small business environment. *Management Research News*, *34*(12), 1334–1350. <https://doi.org/10.1108/01409171111186432>

Magaziner, D. (2022). How your POS system will help your restaurant survive omicron. *Nation's Restaurant News.* <https://www.proquest.com/trade-journals/how-your-pos-system-will-help-restaurant-survive/docview/2621503605/se-2>

Roding, T. (2023). A classification of information-oriented PoS technology from customer perception. <https://www-sciencedirect-com.dlsu.idm.oclc.org/science/article/pii/S0969698922002193?via%3Dihub>

Sourav, A. I., & Emanuel, A. W. R. (2021). Feasibility study of web-based POS system implementation for SMEs in Bangladesh. *IOP Conference Series.Materials Science and Engineering, 1096*(1). <https://doi.org/10.1088/1757-899X/1096/1/012008>

StoreHub. (2021). Top 10 Benefits of a POS System for Your Retail Business. Storehub. Retrieved from <https://www.storehub.com/ph/blog/pos-system-benefits/>

Smith, J. (2022). How Point of Sale (POS) Systems Improve Retail Operations. Entrepreneur. Retrieved from <https://www.entrepreneur.com/article/380388>

# 

# Appendices

## A. User’s Manual

Prerequisites:

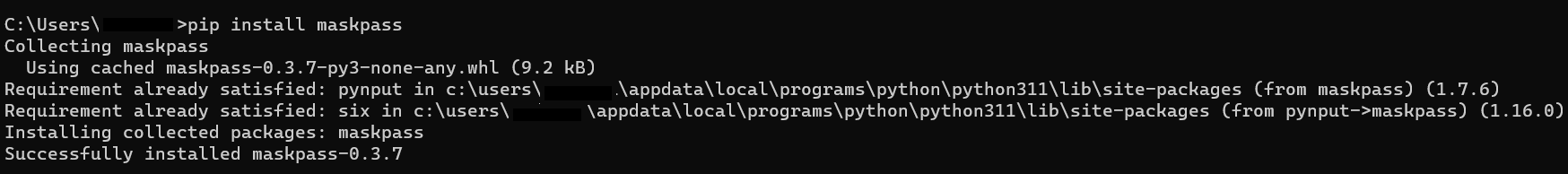
Before running the program, the following python modules should be installed first:

* openpyxl
* maskpass

To install open a command prompt and do the following:

* pip install openpyxl
* pip install maskpass

If successful it must show a “Successfully installed” message



**Running The Code:**

**To run the code just run main.py and create an account**. If you try to login without an account then the code will end with an error message “Create an account first”.

## B. Source Code

**main.py**

from Account import SignIn,SignUp

from menu import Get\_Menu, Display\_Menu

from cart import Cart,find\_cart, checkout, Save\_Cart, Delete\_Cart, Load\_Cart, clear

from receipts import Load\_Transaction, Transaction\_List

import sys

import time

*#main function*

LogIn=0

while LogIn!=1 and LogIn!=2:

print("Log in or Sign up First before accessing the system:")

print("1 - Login")

print("2 - Signup")

*#Login or Sign up Part*

try:

LogIn=int(input(""))

if LogIn==1:

Username=SignIn()

elif LogIn==2:

Username=SignUp()

else:

print("Please enter a valid input")

time.sleep(1)

clear()

*#Catch error when file is not found*

except FileNotFoundError:

with open("accounts.txt","w") as f:

f.write("")

*#catch error when user inputs a string instead of an integer*

except ValueError:

print("Please enter a valid input")

time.sleep(1)

clear()

print("Welcome, "+ Username)

*#get current cart items*

Cart\_Items, Item\_Quantity = [] , []

Cart\_Total=0

shopping\_choice=0

while shopping\_choice!=9:

try:

*#Save\_Cart(Username,"current\_cart",Cart\_Items,Item\_Quantity)*

print("Would you like to:")

print("1 - Go shopping")

print("2 - View your cart")

print("3 - Checkout")

print("4 - Save Cart")

print("5 - View saved carts")

print("6 - Clear Cart")

print("7 - Check your Previous orders")

print("8 - Check your account details")

print("9 - Log out")

shopping\_choice=int(input(""))

clear()

*#1 - GO SHOPPING*

if shopping\_choice==1:

*#gets items from the menu*

item\_id,item\_name,item\_price,item\_description = Get\_Menu()

*#These values make the loops run*

continue\_shopping=1

while continue\_shopping==1:

*#try and except is used to catch errors*

*#if the user inputs a string instead of an integer, it will catch the error*

try:

Display\_Menu(item\_id, item\_name, item\_price, item\_description)

*#getting the item id*

print("Enter the id of the item you wish to buy:", end=" ")

item=int(input(""))

clear()

*#Display Item Details*

print(item\_name[item], end=' - ')

print('P'+str(item\_price[item]))

print("\nDescription:")

print(item\_description[item])

quantity=int(input("Quantity: "))

if item in Cart\_Items:

Item\_Quantity[Cart\_Items.index(item)] += quantity

if quantity!=0 and item not in Cart\_Items:

Item\_Quantity.append(quantity)

Cart\_Items.append(item)

Cart\_Total=Cart(Cart\_Items, Item\_Quantity)

time.sleep(1)

print("\n1 - Continue Shopping")

*#print("2 - View your cart")*

print("2 - Go back to main menu")

*#MENU ENDS HERE IF USER INPUTS anything other than 1*

continue\_shopping=int(input(""))

*#if continue\_shopping==2:*

*# Cart\_Total=Cart(Cart\_Items, Item\_Quantity)*

*# time.sleep(1)*

clear()

*#catch the error with the code below*

except:

print("Please enter a valid input")

time.sleep(1)

*#END OF 1 - GO SHOPPING*

*#2 - CHECK CART*

elif shopping\_choice==2:

if Cart\_Items==[]:

print("Your cart is empty!")

time.sleep(1)

clear()

continue

Cart\_Total=Cart(Cart\_Items, Item\_Quantity)

time.sleep(1)

*#END OF 2 - CHECK CART*

*#3 - CHECKOUT*

elif shopping\_choice==3:

if Cart\_Items==[]:

print("You cannot checkout with an empty cart!")

time.sleep(1)

clear()

continue

Cart\_Items,Item\_Quantity,Cart\_Total=checkout(Username,Cart\_Items,Item\_Quantity, Cart\_Total)

time.sleep(1)

*#4 - SAVE CART*

elif shopping\_choice==4:

if Cart\_Items==[]:

print("Cannot save an empty cart!")

time.sleep(1)

continue

print("Please enter the name of your cart:")

cart\_name=input("")

check=Save\_Cart(Username,cart\_name.lower(),Cart\_Items,Item\_Quantity)

if check ==True:

print("Cart saved!")

time.sleep(1)

else:

print("Cart name already exists! Please delete first")

time.sleep(1)

*#5 - View saved carts*

elif shopping\_choice==5:

while True:

try:

print("1 - View Named Cart")

print("2 - Load Named Cart into Current Cart")

print("3 - Delete Named Cart")

print("0 - Go back to the main menu")

Named\_Cart\_Choice=int(input(""))

if Named\_Cart\_Choice==0:

clear()

break

Cart\_List=Load\_Cart(Username)

if Cart\_List==[]:

print("You have no saved carts")

time.sleep(1)

break

*#View Named Cart*

elif Named\_Cart\_Choice==1:

View\_Cart\_Choice=int(input("Enter the number of the cart you wish to view:\n0 to go back\n"))

if View\_Cart\_Choice==0:

clear()

continue

Cart\_Name=Cart\_List[int(View\_Cart\_Choice)-1]

clear()

View\_Cart\_Items,View\_Item\_Quantity=find\_cart(Username,Cart\_Name)

Cart(View\_Cart\_Items, View\_Item\_Quantity)

time.sleep(2)

*#Load Named Cart into Current Cart*

elif Named\_Cart\_Choice==2:

Load\_Cart\_Choice=int(input("Enter the number of the cart you wish to load:\n0 to go back \n"))

if Load\_Cart\_Choice==0:

clear()

continue

Cart\_Name=Cart\_List[int(Load\_Cart\_Choice)-1]

Cart\_Items,Item\_Quantity=find\_cart(Username,Cart\_Name)

print("Cart loaded!")

time.sleep(1)

clear()

elif Named\_Cart\_Choice==3:

if Cart\_List==[]:

print("You have no saved carts")

time.sleep(1)

clear()

break

view\_cart=int(input("\nEnter the Cart you wish to delete:\n0 to go back\n"))

if view\_cart==0:

clear()

continue

Cart\_Name=Cart\_List[view\_cart-1]

Delete\_Check=Delete\_Cart(Username,Cart\_Name)

if Delete\_Check==True:

print("Cart deleted!")

time.sleep(1)

clear()

else:

print("Please enter a valid input")

time.sleep(1)

clear()

except:

print("Please enter a valid input")

time.sleep(1)

clear()

*#6 - CLEAR CART*

elif shopping\_choice==6:

print("Clearing cart...")

Cart\_Items.clear()

Item\_Quantity.clear()

Cart\_Total = 0

print("Cart cleared!")

time.sleep(1)

clear()

*#7 - CHECK PREVIOUS ORDERS*

elif shopping\_choice==7:

clear()

Transaction\_Number=1

while Transaction\_Number!=0:

try:

Transac\_List=Transaction\_List(Username)

if Transac\_List==[]:

print("You have no previous orders")

time.sleep(1)

clear()

break

print("Transaction List:")

for i in range(len(Transac\_List)):

print(str(i+1)+" - "+Transac\_List[i])

print("Enter the transaction number you wish to view:")

Transaction\_Number=int(input("Input 0 if you wish to go back\n"))

clear()

if Transaction\_Number==0:

break

Transaction=Load\_Transaction(Username,Transac\_List[Transaction\_Number-1])

for i in Transaction:

print(i)

time.sleep(1)

except:

print("Please enter a valid input")

time.sleep(1)

clear()

*#END OF 3 - CHECK PREVIOUS ORDERS*

*#8 - CHECK ACCOUNT DETAILS*

elif shopping\_choice==8:

Account\_Choice=0

while Account\_Choice!=1 or Account\_Choice!=2:

try:

name=""

address=""

contact=""

email=""

*#Get Account Details from the text file*

with open(Username+"\_account.txt","r") as f:

account=f.readlines()

for line in account:

if "Name: " in line:

name=line.split(": ")[1]

elif "Address: " in line:

address=line.split(": ")[1]

elif "Contact Number: " in line:

contact=line.split(": ")[1]

elif "Email: " in line:

email=line.split(": ")[1]

if name!="" or address!="" or contact!="" or email!="":

print("Name: "+name)

print("Address: "+address)

print("Contact Number: "+contact)

print("Email: "+email)

else:

print("No Account Details Found")

print("\n Would you like to:")

print("1 - Edit Account Details")

print("2 - Go Back")

Account\_Choice=int(input(""))

if Account\_Choice==1:

print("Please enter your name: ")

name=input("")

print("Please enter your address: ")

address=input("")

print("Please enter your contact number: ")

contact=input("")

print("Please enter your email address: ")

email=input("")

with open(Username+"\_account.txt","w") as f:

f.writelines("Name: "+name+" \n")

f.writelines("Address: "+address+" \n")

f.writelines("Contact Number: "+contact+" \n")

f.writelines("Email: "+email+ " \n")

print("Shipping details saved!")

time.sleep(1)

break

elif Account\_Choice==2:

break

else:

print("Please enter a valid input")

time.sleep(1)

except FileNotFoundError:

print("No Account Details Found")

Account\_Choice=int(input("1 - Save Account Details\n2 - Go Back\n"))

if Account\_Choice==1:

print("Please enter your name: ")

name=input("")

print("Please enter your address: ")

address=input("")

print("Please enter your contact number: ")

contact=input("")

print("Please enter your email address: ")

email=input("")

with open(Username+"\_account.txt","w") as f:

f.writelines("Name: "+name+" \n")

f.writelines("Address: "+address+" \n")

f.writelines("Contact Number: "+contact+" \n")

f.writelines("Email: "+email+ " \n")

print("Shipping details saved!")

time.sleep(1)

break

elif Account\_Choice==2:

break

else:

print("Please enter a valid input")

time.sleep(1)

except ValueError:

print("Please enter a valid input")

time.sleep(1)

*#END OF 4 - CHECK ACCOUNT DETAILS*

*#9 - LOG OUT*

elif shopping\_choice==9:

print("Logging out...")

print("Thank you for shopping with us!")

print("Come Again!")

sys.exit()

else:

print("Please enter a valid input")

time.sleep(1)

except ValueError:

print("\nPlease enter a valid input \n")

time.sleep(1)

**Cart.py**

**from menu import Get\_Menu**

***#from receipts import Save\_Transaction, Transaction\_List, Load\_Transaction***

**import sys**

**import time**

**import os**

**from receipts import Save\_Transaction, Load\_Transaction, Transaction\_List**

***#displays the cart based on cart items and item quantity***

**def clear():**

**if sys.platform == "win32":**

**os.system("cls")**

**else:**

**os.system("clear")**

**def Cart(Cart\_Items, Item\_Quantity):**

**item\_id,item\_name,item\_price,item\_description = Get\_Menu()**

**Cart\_Total = 0**

**print("---------------------------------------------------------------------")**

**print("\t\t\t\tCART")**

**print("---------------------------------------------------------------------")**

**print("ID \tName\t\t\t\t\t\t\t\tPrice\t\tQuantity")**

**for i in range(len(Cart\_Items)):**

**print(item\_id[Cart\_Items[i]], end='\t')**

**print(item\_name[Cart\_Items[i]].ljust(63), end='')**

**print("P"+str(item\_price[Cart\_Items[i]]), end='\t\t')**

**print(Item\_Quantity[i])**

**Cart\_Total += (item\_price[Cart\_Items[i]]\*Item\_Quantity[i])**

**print("Total: P"+str(Cart\_Total)+"\n")**

**return Cart\_Total**

***#finds the cart of the user***

**def find\_cart(Username,Cart\_Name):**

**try:**

**with open(Username+"\_cart.txt", "r") as f:**

***#begin line is where the item is found***

***#end line is where the item ends***

**begin\_line=0**

**end\_line=0**

**lines=f.readlines()**

***#loop through the txt file the 'i' is the line number***

**for i in range(len(lines)):**

**if Cart\_Name.lower()+":" in lines[i]:**

**begin\_line=i+1**

**if "end\_of\_"+Cart\_Name.lower() in lines[i]:**

**end\_line=i-1**

***#get the items using the begin\_line and end\_line***

**Cart\_Items=[]**

**Item\_Quantity=[]**

**for i in range(begin\_line, end\_line):**

**Cart\_Items.append(int(lines[i].split("\t")[0]))**

**Item\_Quantity.append( int( lines[i].split("\t")[1] ))**

**if Cart\_Items==[]:**

**return [],[]**

***#Find\_Cart returns: Cart\_Items, Item\_Quantity***

**return Cart\_Items, Item\_Quantity**

***#File not found error***

**except FileNotFoundError:**

**print("Cart is empty")**

**return [],[]**

***#checkout part of the shopping cart***

**def checkout(Username, Cart\_Items, Item\_Quantity, Cart\_Total):**

**if Cart\_Items==[]:**

**print("Cannot Checkout with an empty cart!")**

**time.sleep(1)**

**return**

**print("Proceeding to checkout...")**

**shipping\_details\_choice=0**

***#Shipping Details***

**while shipping\_details\_choice !=1 and shipping\_details\_choice !=2:**

**try:**

***#Check first if user has account details saved***

**Account\_Details\_Use=0**

**name=""**

**address=""**

**contact=""**

**email=""**

***#Get Account Details from the text file***

**with open(Username+"\_account.txt","r") as f:**

**account=f.readlines()**

**for line in account:**

**if "Name: " in line:**

**name=line.split(": ")[1]**

**elif "Address: " in line:**

**address=line.split(": ")[1]**

**elif "Contact Number: " in line:**

**contact=line.split(": ")[1]**

**elif "Email: " in line:**

**email=line.split(": ")[1]**

**if name!="" or address!="" or contact!="" or email!="":**

**print("Name: "+name)**

**print("Address: "+address)**

**print("Contact Number: "+contact)**

**print("Email: "+email)**

**Account\_Details\_Use=0**

**while Account\_Details\_Use!=1 and Account\_Details\_Use!=2:**

**print("Would you like to use these details?")**

**print("1 - Yes, 2 - No")**

**Account\_Details\_Use=int(input(""))**

**if Account\_Details\_Use!=1 and Account\_Details\_Use!=2:**

**print("Please enter a valid input!")**

**time.sleep(1)**

**if Account\_Details\_Use==1:**

**break**

**if name=="" or address=="" or contact=="" or email=="":**

**print("No Account Details Found")**

**print("Please enter your name: ")**

**name=input("")**

**print("Please enter your address: ")**

**address=input("")**

**print("Please enter your contact number: ")**

**contact=input("")**

**print("Please enter your email address: ")**

**email=input("")**

***#Ask user if they want to save their shipping details***

**print("Would you like to save your shipping details?")**

**print("1 - Yes, 2 - No")**

***#Save shipping details***

**shipping\_details\_choice=int(input(""))**

**clear()**

***#Username is defined on main function***

**if shipping\_details\_choice==1:**

**with open(Username+"\_account.txt","w") as f:**

**f.writelines("Name: "+name+" \n")**

**f.writelines("Address: "+address+" \n")**

**f.writelines("Contact Number: "+contact+" \n")**

**f.writelines("Email: "+email+ " \n")**

**print("Shipping details saved!")**

***#If user doesn't want to save their shipping details just continue with the code***

***#IF NO FILE IS FOUND ASK USER IF THEY WANT TO SAVE THEIR SHIPPING DETAILS***

***#If user wants to save then save it otherwise just use the info for one transaction***

**except FileNotFoundError:**

**print("No Account Details Found")**

**print("Please enter your name: ")**

**name=input("")**

**print("Please enter your address: ")**

**address=input("")**

**print("Please enter your contact number: ")**

**contact=input("")**

**print("Please enter your email address: ")**

**email=input("")**

**print("\nWould you like to Save your Account Details?\n1 - Yes\n2 - No")**

**shipping\_details\_choice=int(input(""))**

**if shipping\_details\_choice==1:**

**with open(Username+"\_account.txt","w") as f:**

**f.writelines("Name: "+name+" \n")**

**f.writelines("Address: "+address+" \n")**

**f.writelines("Contact Number: "+contact+" \n")**

**f.writelines("Email: "+email+ " \n")**

**print("Shipping details saved!")**

**time.sleep(1)**

**print("Please enter your payment method:")**

**payment=input("")**

**print("Thank you for your purchase!")**

**clear()**

**print("Your order will be delivered to:")**

**print(name)**

**print(address)**

**print(contact)**

**print(email)**

**print("Your payment method is:")**

**print(payment)**

**print("Your total is P"+str(Cart\_Total))**

**Save\_Transaction(Username, Cart\_Items, Item\_Quantity, name, address, contact, email, payment)**

**print("You have successfully placed an Order!")**

**time.sleep(2)**

**clear()**

**return [],[],0**

***#END OF 1 - CHECKOUT***

***#This function saves the cart items***

**def Save\_Cart(Username,Cart\_Name,Cart\_Items,Item\_Quantity):**

**while True:**

**try:**

**with open(Username+"\_cart.txt", "r") as f:**

**lines=f.readlines()**

**for i in lines:**

**if Cart\_Name.lower()+":" in i:**

**return False**

**with open(Username+"\_cart.txt", "a") as f:**

**f.writelines("\n\n")**

**f.writelines(Cart\_Name.lower()+": \n")**

**for i in range(len(Cart\_Items)):**

**f.writelines(str(Cart\_Items[i])+"\t"+str(Item\_Quantity[i])+"\t\n")**

**f.writelines("\nend\_of\_"+Cart\_Name.lower() +" \n")**

**return True**

**except FileNotFoundError:**

**with open(Username+"\_cart.txt", "w") as f:**

**f.writelines("format: \n")**

**f.writelines("item\_id\titem\_quantity\t \n")**

**f.writelines("\nend\_of\_(name of cart) \n")**

**f.writelines("\n")**

**for i in range(len(Cart\_Items)):**

**f.writelines(str(Cart\_Items[i])+"\t"+str(Item\_Quantity[i])+"\t\n")**

**f.writelines("\nend\_of\_current\_cart \n")**

**print("")**

***#loads cart items from the txt file***

**def Load\_Cart(Username):**

**while True:**

**try:**

**with open (Username+"\_cart.txt", "r") as f:**

**lines=f.readlines()**

**Cart\_List=[]**

**for i in lines:**

**if "format" in i or "current\_cart" in i:**

**continue**

**if ":" in i:**

**Cart\_List.append(i.rstrip(": \n"))**

**print("Cart List:")**

**for i in range(len(Cart\_List)):**

**print(str(i+1)+" - "+Cart\_List[i])**

**return Cart\_List**

**except FileNotFoundError:**

**print("No existing Carts, Please create a cart first!")**

**time.sleep(1)**

**return []**

**def Delete\_Cart(Username, Cart\_Name):**

**try:**

**with open(Username+"\_cart.txt", "r") as f:**

***#begin line is where the item is found***

***#end line is where the item ends***

**begin\_line=0**

**end\_line=0**

**lines=f.readlines()**

**ptr=1**

***#loop through the txt file the 'i' is the line number***

**for i in range(len(lines)):**

**if Cart\_Name.lower()+":" in lines[i]:**

**begin\_line=i**

**if "end\_of\_"+Cart\_Name.lower() in lines[i]:**

**end\_line=i+1**

**if begin\_line==0 and end\_line==0:**

**print("Cart not found")**

**return False**

**with open(Username+"\_cart.txt", "w") as fw:**

**for line in lines:**

**if ptr<begin\_line or ptr>end\_line:**

**fw.write(line)**

**ptr+=1**

**return True**

***#File not found error***

**except FileNotFoundError:**

**print("Cart is empty")**

**return [],[]**

**Account.py**

***#maskpass module is used for password masking***

**import maskpass**

**def SignIn():**

**'''function that accepts a username and a password**

**and signs in if the password is correct and exits the program**

**after 5 incorrect attempts**

**'''**

**attempt = 0**

**while attempt<5:**

**try:**

**with open("Accounts.txt","r") as file:**

**file\_read=file.readlines()**

**UsernameIsPresent = False**

**Username = input("Enter your Username: ")**

**Password = maskpass.askpass(prompt="Password:", mask="\*")**

***#check if username is present then check the password after***

**for i in file\_read:**

***#Username is not case sensitive, Password is case sensitive***

**if "."+Username.lower()+" " in i.lower() and ","+Password+" " in i:**

**UsernameIsPresent = True**

**break**

**if UsernameIsPresent == True:**

**return Username**

**elif attempt >= 0 and attempt < 4:**

**print("Invalid Username or Password, Please Try again")**

**attempt+=1**

**else:**

**print("Too many attempts, Please Try again later")**

**exit()**

**except FileNotFoundError:**

**print("File not found, Please create an account first!")**

**attempt=5;**

**exit()**

**def SignUp():**

**"""ffunction that allows users to create an account and saves it into a database**

**"""**

**while True:**

**UsernameIsTaken = False**

**Username=input("Create your username:")**

***#read from database if username exists***

**create\_file=open("Accounts.txt","a")**

**create\_file.writelines("")**

**create\_file.close()**

**file=open("Accounts.txt","r")**

**file\_read=file.readlines()**

***#loop through the database and check if username exists***

**for i in file\_read:**

**if "."+Username.lower()+" " in i.lower():**

**UsernameIsTaken = True**

**break**

**if UsernameIsTaken==True:**

**print("Username is already taken, Please try again")**

**file.close()**

**continue**

**file.close()**

**Password = maskpass.askpass(prompt="Create your Password:", mask="\*")**

**ConfirmPassword = maskpass.askpass(prompt="Confirm your Password:", mask="\*")**

**if Password == ConfirmPassword:**

**file=open("Accounts.txt","a")**

**file.write("."+Username + " ," + Password +" \n")**

**print("Account created successfully")**

**file.close()**

***#append to Accounts.txt***

**return Username**

**else:**

**print("Password does not match, Please try again")**

**continue**

**Menu.py**

**import openpyxl**

***#read excel using***

**dataframe = openpyxl.load\_workbook('itemlist.xlsx')**

**dataframe1 = dataframe.active**

***# Iterate the loop to read the cell values***

***# 1st argument is the row number***

***#Get\_Menu gets the menu from the excel file using openpyxl***

**def Get\_Menu():**

**item\_id=[]**

**item\_name=[]**

**item\_price=[]**

**item\_description=[]**

***#Get the item id in column 1***

**for col in dataframe1.iter\_cols(1, 1):**

**for row in range(1, dataframe1.max\_row):**

**a=col[row].value**

**item\_id.append(a)**

***#get the item name in column 2***

**for col in dataframe1.iter\_cols(2, 2):**

**for row in range(1, dataframe1.max\_row):**

**a=col[row].value**

**item\_name.append(a)**

***#get the item price in column 3***

**for col in dataframe1.iter\_cols(3, 3):**

**for row in range(1, dataframe1.max\_row):**

**a=col[row].value**

**item\_price.append(a)**

***#Get the item description column 4***

**for col in dataframe1.iter\_cols(4, 4):**

***#print(row)***

**for row in range(1, dataframe1.max\_row):**

***#print(col[row].value)***

**a=col[row].value**

**item\_description.append(a)**

**return item\_id, item\_name, item\_price, item\_description**

***#Display\_Menu displays the menu using the item\_id,name,price, and description***

**def Display\_Menu(item\_id, item\_name, item\_price, item\_description):**

**print("---------------------------------------------------------------------")**

**print("\t\t\t\tMENU")**

**print("---------------------------------------------------------------------")**

**print("ID \tName\t\t\t\t\t\t\t\tPrice\t\t")**

**for i in range(len(item\_id)):**

**print(item\_id[i], end='\t')**

**print(item\_name[i].ljust(63), end='')**

**print("P"+str(item\_price[i]))**

**receipts.py**

**import datetime**

**from menu import Get\_Menu**

***#Save\_Transaction saves the transaction in the form of a receipt in a txt file***

**def Save\_Transaction(Username, Cart\_Items, Item\_Quantity, name, address, contact, email, payment):**

**item\_id, item\_name, item\_price, Item\_Description = Get\_Menu()**

**Cart\_Total=0**

**for i in range(len(Cart\_Items)):**

**Cart\_Total += (item\_price[Cart\_Items[i]]\*Item\_Quantity[i])**

**with open(Username.lower()+"\_receipts.txt", "a") as f:**

**f.write("\n------------------------------------------------------------------------------------------------")**

**f.write("\nDate: "+str(datetime.datetime.now())+"")**

**f.write("\nID \t"+"Name".ljust(63)+"Price\t\tQuantity")**

**for i in range(len(Cart\_Items)):**

**f.write("\n"+str(item\_id[Cart\_Items[i]])+"\t"+item\_name[Cart\_Items[i]].ljust(63)+""+str(item\_price[Cart\_Items[i]])+"\t\t"+str(Item\_Quantity[i]))**

**f.write("\nTotal: P"+str(Cart\_Total)+"\n")**

**f.writelines("\nShipping Details:\n")**

**f.write("Name: "+name+"\n")**

**f.write("Address: "+address+"\n")**

**f.write("Contact: "+contact+"\n")**

**f.write("Email: "+email+"\n")**

**f.write("Payment Method: "+payment)**

**f.write("\n------------------------------------------------------------------------------------------------\n")**

***#Load\_Transaction loads a specific transaction from an account using its date***

**def Load\_Transaction(Username, date):**

**try:**

**print("Loading transaction history")**

**with open(Username.lower()+"\_receipts.txt", "r") as f:**

**lines = f.readlines()**

**Begin\_Line=0**

**End\_Line=0**

**Transaction=[]**

**for i in range(len(lines)):**

**if date in lines[i]:**

**Begin\_Line=i-1**

**for i in range(Begin\_Line+1, len(lines)):**

**if "-------" in lines[i]:**

**End\_Line=i**

**break**

**for i in range(Begin\_Line, End\_Line):**

**Transaction.append(lines[i])**

**return Transaction**

**except FileNotFoundError:**

**print("Transaction history file does not exist")**

**print("Creating new file for transaction history")**

**with open(Username.lower()+"\_receipts.txt", "w") as f:**

**f.writelines("")**

***#Transaction\_List Loads the transaction history of a specific account***

**def Transaction\_List(Username):**

**while True:**

**try:**

**Transaction\_List=[]**

**with open(Username.lower()+"\_receipts.txt", "r") as f:**

**lines = f.readlines()**

**for i in lines:**

**if "Date: " in i:**

**i=i.rstrip("\n")**

**Transaction\_List.append(i.split(": ")[1])**

**return Transaction\_List**

**except FileNotFoundError:**

**with open(Username.lower()+"\_receipts.txt", "w") as f:**

**f.writelines("")**

**print("Transaction history file does not exist")**

**print("Creating new file for transaction history")**

## C. Work Breakdown

| Student Name | Task Assigned | Percentage of the Work Contribution |
| --- | --- | --- |
| Fernandez, Edge Matthew | Project Documentation:  Background of the Study  Review of Related Literature  Conceptual Framework  Pseudocode  Code:  Checked the Code | 30% |
| Enriquez, John Paolo | Project Documentation:  Background of the Study  Objectives  Significance of the Study  Review of Related Literature  Conceptual Framework  Discussion of Results  Code:  Menu Excel Sheet  Checked the Code | 30% |
| EVALLA, Christian Matthew G. | Project Documentation:  Introduction  Background of the Study  Problem Statement  Review of Related Literature  Conceptual Framework  Hierarchy Chart  Flowchart  Results  User’s Manual  Source Codes  Code:  Made the Code | 40% |

## D. Personal Data Sheet



John Paolo Enriquez - A Computer Engineering student ID 122, who is fascinated by how computers are made. I believe that computer engineering is the course that will satisfy my desired future career.



Christian Matthew Evalla - A Computer Engineering student ID 122, who loves learning more about computers. I chose computer engineering as my course because I find it fun to learn about technology, especially about computers.

Edge Matthew Fernandez - A Computer Engineering student ID no. 122, who dreams to become a business owner and entrepreneur with the help of the principles he is learning in college. I chose computer engineering as my course because of the fascination I took from building and repairing computers. I foresee that my stay in this course will aid my abilities in managing and fulfilling my dream.