BSc (Hons) Artificial Intelligence and Data Science

Module: CM1601 Programming Fundamentals

Coursework 1 Report

Module Leader: Ms. Sachinthani Perera

RGU Student ID : 2311130

IIT Student ID : 20222408

Student Name : Vanuja Thihansith Sooriyaarachchi

Acknowledgment

It is always a pleasure to remind the fine people in our IIT Institute for their sincere guidance I received throughout the course work of the module Introduction to IT.

Firstly, I would like to thank Ms. Sachinthani Perera, the module leader of 'Programming Fundamentals.' for her exemplary guidance, constant encouragement, and the careful monitoring throughout the lessons.

Secondly, I would like to thank Mr. Dinusha Kumara the tutorial lecturer of the module who guided us throughout the lectures.

Next, I would like to award a special thanks to my parents and family members for their love, encouragement, and guidance they gave me throughout my coursework activity.

Finally, I would like to thank all my friends, teachers and all others who even helped me from a word to get a success in my project. All your encouragements when the times got rough are much appreciated and noted. So once again, I would like to give my heartiest thanks to all for helping and guiding me to get success in this project.

Table of Content

List of Figures

List of Tables

Problem

You are instructed to create a command line application for the following scenario using Python.

Internet Cafes has become a popular business due to the rapid change in the technology and higher demand. Many people, especially students who do not have access to a personal computer, go to internet cafes for completing their projects and assignments, and for entertainment.

John is planning to start an Internet cafe named 'One Net Cafe' in his hometown so that people without personal computer can use the facilitates. He has bought some of the items needed to the cafe and looking for suppliers to buy the rest. He has already received details of six wholesale suppliers and each one is selling items with good quality with low prices. When researched, the quotations are almost same. Due the similarities in the suppliers, John has decided to select four dealers randomly to purchase the remaining products. As high number of equipment's required for the café, John need a system to manage inventory.

System should display the following console menu when launched.

- Type AID for adding item details.
- Type DID for deleting item details.
- Type UID for updating item details.
- Type VID for viewing the items table. (Sort according to the items category) and print the current total.
- Type SID for saving the item details to the text file at any time.
- Type SDD for selecting four dealers randomly from a file.
- Type VRL for displaying all the details of the randomly selected dealers. (Sorted according to the location.)
- Type LDI for display the items of the given dealer.
- Type ESC to exit the program.

Python Codes

```
import random

# insert the data in a table format
from tabulate import tabulate

print("\n*****Welcome to One Net Cafe Inventory Management System*****\n")
# dictionary to store item details
items = []


# function to display the console menu
def menu():
    print("Type AID for adding item details.")
    print("Type DID for deleting item details.")
    print("Type UID for updating item details.")
    print("Type VID for viewing the items table.")
    print("Type SID for saving the item details to the text file at any
time.")
    print("Type SDD for selecting four dealers randomly from a file.")
    print("Type VRL for displaying all the details of the randomly selected
dealers.")
    print("Type LDI for display the items of the given dealer.")
    print("Type ESC to exit the program.")


menu()

inventory = []
random_dealers = []
Duplicate_code = []

while True:
    choice = input("Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL,
LDI, ESC : ")

    if choice.upper() == "AID":
        # define a list to store items in the inventory


        def add_item():
            print("Enter the item details.")
            # Handle the exception as desired
            while True:
                try:
                    item_code = int(input("Item Code: "))
                    if item_code < 0:
                        print("Please Enter A Positive Number!")
                    elif item_code in Duplicate_code:
                        print("Duplicate found. Enter a another Item Code.")
                    else:
                        Duplicate_code.append(item_code)
                    break
                except ValueError:
                    print("Please Enter A Valid Code!")
```

```
            while True:
                try:
                    item_name = input("Item Name: ")
                    if item_name == "":
                        raise ValueError("Item Name Cannot Be Empty!")
                    break
                except ValueError as ve:
                    print(ve)
            while True:
                try:
                    item_brand = input("Item Brand: ")
                    if item_brand == "":
                        raise ValueError("Item Brand Cannot Be Empty!")
                    break
                except ValueError as ve:
                    print(ve)
            while True:
                try:
                    item_price = float(input("Item Price: "))
                    if item_price < 0:
                        print("Price Cannot Be Negative!")
                    break
                except ValueError:
                    print("Please Enter A Valid Price!")
            while True:
                try:
                    item_quantity = int(input("Item Quantity: "))
                    if item_quantity < 0:
                        print("Quantity Cannot Be Negative!")
                    break
                except ValueError:
                    print("Please Enter A Valid Quantity!")
            while True:
                try:
                    item_category = input("Item Category: ")
                    if item_category == "":
                        raise ValueError("Item Category Cannot Be Empty!")
                    break
                except ValueError as ve:
                    print(ve)
            while True:
                try:
                    purchased_date = str(input("Purchased Date (YYYY-MM-DD):
"))
                    break
                except ValueError:
                    print("Please Enter A Valid Date!")
            item = {
                "item_code": item_code,
                "item_name": item_name,
                "item_brand": item_brand,
                "item_price": item_price,
                "item_quantity": item_quantity,
                "item_category": item_category,
                "purchased_date": purchased_date
            }
            inventory.append(item)
```

```
            print(inventory)
            print("Item Added Successfully!")


        add_item()

    elif choice.upper() == "DID":

        # function to delete item details from the system
        def delete_item():
            while True:
                try:
                    item_code = int(input("Enter Item Code: "))
                    if item_code < 0:
                        print("Please Enter A Positive Number!")
                        delete_item()
                    break
                except ValueError:
                    print("Please Enter A Valid Code!")
            for item in inventory:
                if item["item_code"] == item_code:
                    inventory.remove(item)
                    print("Item Deleted!")
                    break
            else:
                print("This Item Code NOT Found In The System!")


        delete_item()

    elif choice.upper() == "UID":

        # function to update item details in the system
        def update_item():
            print("Updating an item...")
            item_code = int(input("Enter item code: "))
            for item in inventory:
                if item["item_code"] == item_code:
                    print("Update Details: ")
                    while True:
                        try:
                            item_name = input("Item Name: ")
                            if item_name == "":
                                raise ValueError("Item Name Cannot Be
Empty!")
                            break
                        except ValueError as ve:
                            print(ve)
                    while True:
                        try:
                            item_brand = input("Enter item brand: ")
                            if item_brand == "":
                                raise ValueError("Item Brand Cannot Be
Empty!")
                            break
                        except ValueError as ve:
                            print(ve)
```

```
                    while True:
                        try:
                            item_price = float(input("Enter price: "))
                            if item_price < 0:
                                print("Price Cannot Be Negative!")
                            break
                        except ValueError:
                            print("Please Enter A Valid Price!")
                    while True:
                        try:
                            quantity = int(input("Enter quantity: "))
                            if quantity < 0:
                                print("Quantity Cannot Be Negative!")
                            break
                        except ValueError:
                            print("Please Enter A Valid Quantity!")
                    while True:
                        try:
                            category = input("Enter category: ")
                            if category == "":
                                raise ValueError("Category Cannot Be Empty!")
                            break
                        except ValueError as ve:
                            print(ve)
                    while True:
                        try:
                            purchased_date = str(input("Enter date (YYYY-MM-
DD): "))
                            break
                        except ValueError:
                            print("Please Enter A Valid Date!")
                    item["item_name"] = item_name
                    item["item_brand"] = item_brand
                    item["item_price"] = item_price
                    item["item_quantity"] = quantity
                    item["item_category"] = category
                    item["purchased_date"] = purchased_date
                    print(inventory)
                    print("Item Updated!")
                    break

            else:
                print("Item NOT Found!")

        update_item()

    elif choice.upper() == "VID":
        print(inventory)

        # function to view item details in the system
        def view_item():
            # bubble sort the inventory by item_id
            n = len(inventory)
            for i in range(n):
                for j in range(0, n - i - 1):
                    if inventory[j]['item_code'] < inventory[j +
1]['item_code']:
```

```
                        inventory[j], inventory[j + 1] = inventory[j + 1],
inventory[j]
            # function to view item details in the system
            print(tabulate(inventory, headers="keys", tablefmt="simple_grid",
numalign="right"))
        view_item()

    elif choice.upper() == "SID":
        # function to save item details in the text file
        def save_item():
            with open("save_item_data.txt", "w") as f:
                for i in inventory:
                    f.write(str(i))
                    print("Item Saved!")
        save_item()

    elif choice.upper() == "SDD":
        # function to select dealers randomly from text file
        def select_dealers():
            global random_dealers
        with open("dealers.txt", "r") as f:
            # select only 4 dealers
              lines = f.readlines()
              while len(random_dealers) < 4:
                  dealer = random.choice(lines)
                  dealer_name = dealer.split(",")[0]

                # check if the dealer name is already in the list
                  if dealer_name not in [d[0] for d in random_dealers]:
                    random_dealers.append((dealer_name, dealer))
                    print(dealer_name)
              print("4 dealers are selected randomly")

        select_dealers()

    elif choice.upper() == "VRL":
        # function to view dealer's details
        def dealers_details(dealers):
            # Declare a global variable to access outside the function
            global random_dealers
            # Sort the dealers list based on the second element of each tuple
using a lambda function
            sorted_dealers = sorted(dealers, key=lambda x: x[1])
            return sorted_dealers

        # Check if the global variable 'random_dealers' is empty
        if not random_dealers:
            print("Please Select the dealer's first.")

        sorted_dealers = dealers_details(random_dealers)

        for dealer in sorted_dealers:
            print(f"Dealer: {dealer[0]}")
            print(f"Details: {dealer[1]}")
            print("_" * 20)

    elif choice.upper() == "LDI":
```

```
        # function to show items from randomly selected dealers
        def display_items():
            with open("items.txt", "r") as f:
                lines = f.readlines()

            dealers_items = {}

            # Iterate through each line of input
            for line in lines:
                # Split each line into its four parts
                item, dealers, price, quantity = line.strip().split(", ")
                if dealers.lower() not in dealers_items:
                    dealers_items[dealers.lower()] = []
                dealers_items[dealers.lower()].append((item, price,
quantity))

            dealers = []

            for i in range(4):
                dealer_name = input("Enter dealer name {} of 4: ".format(i +
1))
                while dealer_name.lower() not in dealers_items:
                    print("Dealer not found. Please try again.")
                    dealer_name = input("Enter dealer name {} of 4:
".format(i + 1))
                dealers.append(dealer_name.lower())

            print("\n-----------------------------------------------------
\n")

            for dealer_name in dealers:
                # Display the dealer's name with the first letter capitalized
                print("Dealer Name: {}".format(dealer_name.capitalize()))
                for item in dealers_items[dealer_name]:
                    print("Item: {}, Price: {}, Quantity: {}".format(item[0],
item[1], item[2]))
                print()
        display_items()

    elif choice.upper() == "ESC":
        # function to terminate program
        print("\n*** Thank You! ***\n")
        break

    else:
        # to display the menu options and prompt the user to enter a valid
input.
        print("\n*Invalid Input.Try again*\n")
        menu()
```

```
import random


# insert the data in a table format
from tabulate import tabulate


print("\n*****Welcome to One Net Cafe Inventory Management System*****\n")
# dictionary to store item details
items = []



# function to display the console menu
def menu():
    print("Type AID for adding item details.")
    print("Type DID for deleting item details.")
    print("Type UID for updating item details.")
    print("Type VID for viewing the items table.")
    print("Type SID for saving the item details to the text file at any time.")
    print("Type SDD for selecting four dealers randomly from a file.")
    print("Type VRL for displaying all the details of the randomly selected dealers.")
    print("Type LDI for display the items of the given dealer.")
    print("Type ESC to exit the program.")



menu()


inventory = []
random_dealers = []
Duplicate_code = []
```

```
while True:
    choice = input("Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : ")

    if choice.upper() == "AID":
        # define a list to store items in the inventory



        def add_item():
            print("Enter the item details.")
            # Handle the exception as desired
            while True:
                try:
                    item_code = int(input("Item Code: "))
                    if item_code < 0:
                        print("Please Enter A Positive Number!")
                    elif item_code in Duplicate_code:
                        print("Duplicate found. Enter a another Item Code.")
                    else:
                        Duplicate_code.append(item_code)
                    break
                except ValueError:
                    print("Please Enter A Valid Code!")
            while True:
                try:
                    item_name = input("Item Name: ")
                    if item_name == "":
                        raise ValueError("Item Name Cannot Be Empty!")
```

```
            break
        except ValueError as ve:
            print(ve)
    while True:
        try:
            item_brand = input("Item Brand: ")
            if item_brand == "":
                raise ValueError("Item Brand Cannot Be Empty!")
            break
        except ValueError as ve:
            print(ve)
    while True:
        try:
            item_price = float(input("Item Price: "))
            if item_price < 0:
                print("Price Cannot Be Negative!")
            break
        except ValueError:
            print("Please Enter A Valid Price!")
    while True:
        try:
            item_quantity = int(input("Item Quantity: "))
            if item_quantity < 0:
                print("Quantity Cannot Be Negative!")
            break
        except ValueError:
            print("Please Enter A Valid Quantity!")
    while True:
```

```
        try:
            item_category = input("Item Category: ")
            if item_category == "":
                raise ValueError("Item Category Cannot Be Empty!")
            break
        except ValueError as ve:
            print(ve)
    while True:
        try:
            purchased_date = str(input("Purchased Date (YYYY-MM-DD): "))
            break
        except ValueError:
            print("Please Enter A Valid Date!")
    item = {
        "item_code": item_code,
        "item_name": item_name,
        "item_brand": item_brand,
        "item_price": item_price,
        "item_quantity": item_quantity,
        "item_category": item_category,
        "purchased_date": purchased_date
    }
    inventory.append(item)
    print(inventory)
    print("Item Added Successfully!")


add_item()
```

```
elif choice.upper() == "DID":

    # function to delete item details from the system
    def delete_item():
        while True:
            try:
                item_code = int(input("Enter Item Code: "))
                if item_code < 0:
                    print("Please Enter A Positive Number!")
                    delete_item()
                break
            except ValueError:
                print("Please Enter A Valid Code!")
        for item in inventory:
            if item["item_code"] == item_code:
                inventory.remove(item)
                print("Item Deleted!")
                break
        else:
            print("This Item Code NOT Found In The System!")



    delete_item()

elif choice.upper() == "UID":

    # function to update item details in the system
```

```
def update_item():
    print("Updating an item...")
    item_code = int(input("Enter item code: "))
    for item in inventory:
        if item["item_code"] == item_code:
            print("Update Details: ")
            while True:
                try:
                    item_name = input("Item Name: ")
                    if item_name == "":
                        raise ValueError("Item Name Cannot Be Empty!")
                    break
                except ValueError as ve:
                    print(ve)
            while True:
                try:
                    item_brand = input("Enter item brand: ")
                    if item_brand == "":
                        raise ValueError("Item Brand Cannot Be Empty!")
                    break
                except ValueError as ve:
                    print(ve)
            while True:
                try:
                    item_price = float(input("Enter price: "))
                    if item_price < 0:
                        print("Price Cannot Be Negative!")
                    break
```

```
        except ValueError:
            print("Please Enter A Valid Price!")
    while True:
        try:
            quantity = int(input("Enter quantity: "))
            if quantity < 0:
                print("Quantity Cannot Be Negative!")
            break
        except ValueError:
            print("Please Enter A Valid Quantity!")
    while True:
        try:
            category = input("Enter category: ")
            if category == "":
                raise ValueError("Category Cannot Be Empty!")
            break
        except ValueError as ve:
            print(ve)
    while True:
        try:
            purchased_date = str(input("Enter date (YYYY-MM-DD): "))
            break
        except ValueError:
            print("Please Enter A Valid Date!")
    item["item_name"] = item_name
    item["item_brand"] = item_brand
    item["item_price"] = item_price
    item["item_quantity"] = quantity
```

```
            item["item_category"] = category
            item["purchased_date"] = purchased_date
            print(inventory)
            print("Item Updated!")
            break

        else:
            print("Item NOT Found!")

    update_item()

elif choice.upper() == "VID":
    print(inventory)

    # function to view item details in the system
    def view_item():
        # bubble sort the inventory by item_id
        n = len(inventory)
        for i in range(n):
            for j in range(0, n - i - 1):
                if inventory[j]['item_code'] < inventory[j + 1]['item_code']:
                    inventory[j], inventory[j + 1] = inventory[j + 1], inventory[j]
        # function to view item details in the system
        print(tabulate(inventory, headers="keys", tablefmt="simple_grid", numalign="right"))
    view_item()

elif choice.upper() == "SID":
    # function to save item details in the text file
```

```
    def save_item():
        with open("save_item_data.txt", "w") as f:
            for i in inventory:
                f.write(str(i))
                print("Item Saved!")
    save_item()

elif choice.upper() == "SDD":
    # function to select dealers randomly from text file
    def select_dealers():
        global random_dealers
    with open("dealers.txt", "r") as f:
        # select only 4 dealers
          lines = f.readlines()
          while len(random_dealers) < 4:
              dealer = random.choice(lines)
              dealer_name = dealer.split(",")[0]

            # check if the dealer name is already in the list
              if dealer_name not in [d[0] for d in random_dealers]:
                random_dealers.append((dealer_name, dealer))
                print(dealer_name)
          print("4 dealers are selected randomly")

    select_dealers()

elif choice.upper() == "VRL":
    # function to view dealer's details
```

```
    def dealers_details(dealers):
        # Declare a global variable to access outside the function
        global random_dealers
        # Sort the dealers list based on the second element of each tuple using a lambda function
        sorted_dealers = sorted(dealers, key=lambda x: x[1])
        return sorted_dealers

    # Check if the global variable 'random_dealers' is empty
    if not random_dealers:
        print("Please Select the dealer's first.")

    sorted_dealers = dealers_details(random_dealers)

    for dealer in sorted_dealers:
        print(f"Dealer: {dealer[0]}")
        print(f"Details: {dealer[1]}")
        print("_" * 20)

elif choice.upper() == "LDI":
    # function to show items from randomly selected dealers
    def display_items():
        with open("items.txt", "r") as f:
            lines = f.readlines()

        dealers_items = {}

        # Iterate through each line of input
        for line in lines:
```

```
            # Split each line into its four parts
            item, dealers, price, quantity = line.strip().split(", ")
            if dealers.lower() not in dealers_items:
                dealers_items[dealers.lower()] = []
            dealers_items[dealers.lower()].append((item, price, quantity))

        dealers = []

        for i in range(4):
            dealer_name = input("Enter dealer name {} of 4: ".format(i + 1))
            while dealer_name.lower() not in dealers_items:
                print("Dealer not found. Please try again.")
                dealer_name = input("Enter dealer name {} of 4: ".format(i + 1))
            dealers.append(dealer_name.lower())

        print("\n-\n" * 20)

        for dealer_name in dealers:
            # Display the dealer's name with the first letter capitalized
            print("Dealer Name: {}".format(dealer_name.capitalize()))
            for item in dealers_items[dealer_name]:
                print("Item: {}, Price: {}, Quantity: {}".format(item[0], item[1], item[2]))
            print()
    display_items()

elif choice.upper() == "ESC":
    # function to terminate program
    print("\n*** Thank You! ***\n")
```

```
    break


else:
    # to display the menu options and prompt the user to enter a valid input.
    print("\n*Invalid Input.Try again*\n")
    menu()
```

External Packages Used

- Tabulate package

  Tabulate is an open-source python package that can be used to insert the data in a table format.

  1. Open PyCharm project.
  2. Click on “File” in the top menu bar, then click on “Settings”.
  3. In the “Settings” window, click on “Project: <your_project_name>” on the left-hand side.
  4. Click on “Project Interpreter”.
  5. Click on the “+” button to add a new package.
  6. In the search bar, type “Tabulate”.
  7. Select “Tabulate” from the search results and click on the “Install Package” button.
  8. Wait for PyCharm to download and install the “Tabulate” package.
  9. Once the installation is complete, you can start using the “Tabulate” library in your Python code.

Table of test cases used to test the programs and the results.

- Console Menu >> Type AID for adding item details >> Display item detail's inputs (Item Code, Item Name, Item Brand, Item Price, Item Quantity, Item Category, Purchased Date)

Test Case	Inputs	Expected output	Actual output	Remark
1	Select an option (AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC): >> AID (Selecting "Adding item details") Item Code: 01 Item Name: Speaker Item Brand: Abans Item Price: 5000 Item Quantity: 5 Item Category: Sound Purchased Date (YYYY-MM-DD): 2023.03.04	[{'item_code': 1, 'item_name': 'Speaker', 'item_brand': 'Abans', 'item_price': 5000.0, 'item_quantity': 5, 'item_category': 'Sound', 'purchased_date': '2023.03.04'}] Item Added Successfully! ---------- Displaying item details	[{'item_code': 1, 'item_name': 'Speaker', 'item_brand': 'Abans', 'item_price': 5000.0, 'item_quantity': 5, 'item_category': 'Sound', 'purchased_date': '2023.03.04'}] Item Added Successfully! ---------- Displaying item details	Pass

Table 1 – Test case 1

```
*****Welcome to One Net Cafe Inventory Management System*****

Type AID for adding item details.
Type DID for deleting item details.
Type UID for updating item details.
Type VID for viewing the items table.
Type SID for saving the item details to the text file at any time.
Type SDD for selecting four dealers randomly from a file.
Type VRL for displaying all the details of the randomly selected dealers.
Type LDI for display the items of the given dealer.
Type ESC to exit the program.
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : aid
Enter the item details.
Item Code: 01
Item Name: Speaker
Item Brand: Abans
Item Price: 5000
Item Quantity: 5
Item Category: Sound
Purchased Date (YYYY-MM-DD): 2023.03.04
[{'item_code': 1, 'item_name': 'Speaker', 'item_brand': 'Abans', 'item_price': 5000.0, 'item_quantity': 5, 'item_category': 'Sound', 'purchased_date': '2023.03.04'}]
Item Added Successfully!
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC :
```

Figure 1 – Adding item details.

- Console menu >> Type DID for deleting item details >> Ask item code

Test Case	Inputs	Expected output	Actual output	Remark
2	Select an option (AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC): >> DID (Selecting “Deleting item details”) Enter Item Code: 01	Item Deleted! ---------- Ask item code and delete it	Item Deleted! ---------- Ask item code and delete it	Pass

Table 2 – Test case 2

```
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : did
Enter Item Code: 01
Item Deleted!
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC :
```

Figure 2 – Deleting item details.

- Console menu >> Type UID for updating item details >> Ask item code

Test Case	Inputs	Expected output	Actual output	Remark
3	Select an option (AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC): >> UID (Selecting "Updating item details") Enter item code: 02 Item name: Keyboard Enter item brand: Nanotech Enter price: 2700 Enter quantity: 10 Enter category: PC Enter date (YYYY-MM-DD): 2023.03.07	[{'item_code': 2, 'item_name': 'Keyboard', 'item_brand': 'Nanotech', 'item_price': 2700.0, 'item_quantity': 10, 'item_category': 'PC', 'purchased_date': '2023.03.07'}] ---------- Displaying the list with updated item details	[{'item_code': 2, 'item_name': 'Keyboard', 'item_brand': 'Nanotech', 'item_price': 2700.0, 'item_quantity': 10, 'item_category': 'PC', 'purchased_date': '2023.03.07'}] ---------- Displaying the list with updated item details	Pass

Table 3 – Test case 3

```
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : uid
Enter the item details.
Item Code: 02
Item Name: Keyboard
Item Brand: Singer
Item Price: 2500
Item Quantity: 10
Item Category: PC
Purchased Date (YYYY-MM-DD): 2023.03.06
[{'item_code': 2, 'item_name': 'Keyboard', 'item_brand': 'Singer', 'item_price': 2500.0, 'item_quantity': 10, 'item_category': 'PC', 'purchased_date': '2023.03.06'}]
Item Added Successfully!
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : uid
Updating an item...
Enter item code: 02
Update Details:
Item Name: Keyboard
Enter item brand: Nanotech
Enter price: 2700
Enter quantity: 10
Enter category: PC
Enter date (YYYY-MM-DD): 2023.03.07
[{'item_code': 2, 'item_name': 'Keyboard', 'item_brand': 'Nanotech', 'item_price': 2700.0, 'item_quantity': 10, 'item_category': 'PC', 'purchased_date': '2023.03.07'}]
Item Updated!
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC :
```

Figure 3 – Updating item details.

- Console menu >> Type VID for view item details >> Display item details in a table format

Test Case	Inputs	Expected output	Actual output	Remarks
4	Select an option (AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC): >> VID (Selecting “View item details”)	Display item details as table with sorted item codes	Display item details as table with sorted item codes	Pass

Table 4 – Test case 4

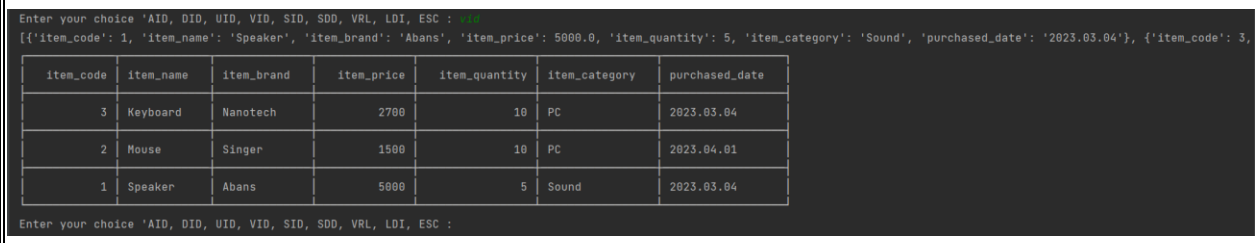

Figure 4 – View item details.

- Console menu >> Type SID for saving item details to the text file >> Item details saving as dictionary in text file.

Test Case	Inputs	Expected output	Actual output	Remarks
5	Select an option (AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC): >> SID (Selecting "Save item details")	{'item_code': 3, 'item_name': 'Keyboard', 'item_brand': 'Nanotech', 'item_price': 2700.0, 'item_quantity': 10, 'item_category': 'PC', 'purchased_date': '2023.03.04'}{'item_code': 2, 'item_name': 'Mouse', 'item_brand': 'Singer', 'item_price': 1500.0, 'item_quantity': 10, 'item_category': 'PC', 'purchased_date': '2023.04.01'}{'item_code': 1, 'item_name': 'Speaker', 'item_brand': 'Abans', 'item_price': 5000.0, 'item_quantity': 5, 'item_category': 'Sound', 'purchased_date': '2023.03.04'} The added items are saved as a dictionary in a text file called "save_item_data.txt"	{'item_code': 3, 'item_name': 'Keyboard', 'item_brand': 'Nanotech', 'item_price': 2700.0, 'item_quantity': 10, 'item_category': 'PC', 'purchased_date': '2023.03.04'}{'item_code': 2, 'item_name': 'Mouse', 'item_brand': 'Singer', 'item_price': 1500.0, 'item_quantity': 10, 'item_category': 'PC', 'purchased_date': '2023.04.01'}{'item_code': 1, 'item_name': 'Speaker', 'item_brand': 'Abans', 'item_price': 5000.0, 'item_quantity': 5, 'item_category': 'Sound', 'purchased_date': '2023.03.04'} The added items are saved as a dictionary in a text file called "save_item_data.txt"	Pass

Table 5 – Test case 5

```
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : sid
Item Saved!
Item Saved!
Item Saved!
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC :
```

Figure 5 – Save item details.

- Console menu >> Type SDD for selecting four dealers randomly from a file >> Display random 4 dealers

Test Case	Inputs	Expected output	Actual output	Remarks
6	Select an option (AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC): >> SDD (Selecting 4 dealers randomly)	Singer Nanotech Abans LibertyPlaza ---------- Display 4 different random dealers from text file called "dealers.txt"	Singer Nanotech Abans LibertyPlaza ---------- Display 4 different random dealers from text file called "dealers.txt"	Pass

Table 6 – Test case 6

```
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : sdd
Singer
Nanotech
Abans
LibertyPlaza
4 dealers are selected randomly
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC :
```

Figure 6 – Select random dealers.

- Console menu >> Type VRL for displaying all the details of the randomly selected dealers >> Display dealer name, dealer contact and dealer location

Test Case	Inputs	Expected output	Actual output	Remarks
7	Select an option (AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC): >> VRL (Displaying all the details of the randomly selected dealers)	Dealer: Abans Details: Abans, 077 552 4356, Maharagama _____ Dealer: LibertyPlaza Details: LibertyPlaza, 074 234 5326, Bambalapitiya _____ Dealer: Nanotech Details: Nanotech, 077 445 2323, Piliyandala _____ Dealer: Singer Details: Singer, 071 567 6677, Kottawa _____ Displaying Dealer name, contact and location	Dealer: Abans Details: Abans, 077 552 4356, Maharagama _____ Dealer: LibertyPlaza Details: LibertyPlaza, 074 234 5326, Bambalapitiya _____ Dealer: Nanotech Details: Nanotech, 077 445 2323, Piliyandala _____ Dealer: Singer Details: Singer, 071 567 6677, Kottawa _____ Displaying Dealer name, contact and location	Pass

Table 7 – Test case 7

```
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : vrl
Dealer: Abans
Details: Abans, 077 552 4356, Maharagama


--------------------
Dealer: LibertyPlaza
Details: LibertyPlaza, 074 234 5326, Bambalapitiya
--------------------
Dealer: Nanotech
Details: Nanotech, 077 445 2323, Piliyandala


--------------------
Dealer: Singer
Details: Singer, 071 567 6677, Kottawa


--------------------
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC :
```

Figure 7 – Displaying details of the selected random dealers.

- Console menu >> Type LDI for displaying the item of the given dealer >> Displaying dealer name and 3 items for each dealer

Test Case	Inputs	Expected output	Actual output	Remarks
8	Select an option (AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC): >> LDI (Display the items of the given dealer) Enter dealer name 1 of 4: abans Enter dealer name 2 of 4: libertyplaza Enter dealer name 3 of 4: softlogic Enter dealer name 4 of 4: nanotech	Dealer Name: Abans Item: Monitor, Price: 100000, Quantity: 5 Item: RAM, Price: 8000, Quantity: 5 Item: Graphic Card, Price: 60000, Quantity: 7 Dealer Name: Libertyplaza Item: SSD, Price: 15000, Quantity: 6 Item: ComputerTable, Price: 7000, Quantity: 4 Item: RGB, Price: 3500, Quantity: 20 Dealer Name: Softlogic Item: VGA Cable, Price: 2000, Quantity: 7 Item: MotherBoard, Price: 50000, Quantity: 10 Item: Casing, Price: 5000, Quantity: 10 Dealer Name: Nanotech Item: UPS, Price: 8000, Quantity: 5 Item: HardDisk, Price: 9000, Quantity: 8 Item: PenDrive, Price: 3000, Quantity: 20 Displaying dealer name and their item detais	Dealer Name: Abans Item: Monitor, Price: 100000, Quantity: 5 Item: RAM, Price: 8000, Quantity: 5 Item: Graphic Card, Price: 60000, Quantity: 7 Dealer Name: Libertyplaza Item: SSD, Price: 15000, Quantity: 6 Item: ComputerTable, Price: 7000, Quantity: 4 Item: RGB, Price: 3500, Quantity: 20 Dealer Name: Softlogic Item: VGA Cable, Price: 2000, Quantity: 7 Item: MotherBoard, Price: 50000, Quantity: 10 Item: Casing, Price: 5000, Quantity: 10 Dealer Name: Nanotech Item: UPS, Price: 8000, Quantity: 5 Item: HardDisk, Price: 9000, Quantity: 8 Item: PenDrive, Price: 3000, Quantity: 20 Displaying dealer name and their item detais	Pass

Table 8 – Test case 8

```
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : ldi
Enter dealer name 1 of 4: abans
Enter dealer name 2 of 4: libertyplaza
Enter dealer name 3 of 4: softlogic
Enter dealer name 4 of 4: nanotech

--------------------------------------------------

Dealer Name: Abans
Item: Monitor, Price: 100000, Quantity: 5
Item: RAM, Price: 8000, Quantity: 5
Item: Graphic Card, Price: 60000, Quantity: 7

Dealer Name: Libertyplaza
Item: SSD, Price: 15000, Quantity: 6
Item: ComputerTable, Price: 7000, Quantity: 4
Item: RGB, Price: 3500, Quantity: 20

Dealer Name: Softlogic
Item: VGA Cable, Price: 2000, Quantity: 7
Item: MotherBoard, Price: 50000, Quantity: 10
Item: Casing, Price: 5000, Quantity: 10

Dealer Name: Nanotech
Item: UPS, Price: 8000, Quantity: 5
Item: HardDisk, Price: 9000, Quantity: 8
Item: PenDrive, Price: 3000, Quantity: 20

Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC :
```

Figure 8 – Displaying the item of the given dealer.

- Console menu >> Type ESC for terminate the program >> Terminate the program

Test Case	Inputs	Expected output	Actual output	Remarks
9	Select an option (AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC): >> ESC (Terminating the program)	*** Thank You! *** Terminate the program	*** Thank You! *** Terminate the program	Pass

Table 9 – Test case 9

```
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : esc

*** Thank You! ***


Process finished with exit code 0
```

Figure 9 – Terminate the program.

Error Handling

```
*****Welcome to One Net Cafe Inventory Management System*****

Type AID for adding item details.
Type DID for deleting item details.
Type UID for updating item details.
Type VID for viewing the items table.
Type SID for saving the item details to the text file at any time.
Type SDD for selecting four dealers randomly from a file.
Type VRL for displaying all the details of the randomly selected dealers.
Type LDI for display the items of the given dealer.
Type ESC to exit the program.
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : 1

*Invalid Input.Try again*

Type AID for adding item details.
Type DID for deleting item details.
Type UID for updating item details.
Type VID for viewing the items table.
Type SID for saving the item details to the text file at any time.
Type SDD for selecting four dealers randomly from a file.
Type VRL for displaying all the details of the randomly selected dealers.
Type LDI for display the items of the given dealer.
Type ESC to exit the program.
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC :
```

Figure 10 – Error Handling Figure 1

```
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : aid
Enter the item details.
Item Code: sd
Please Enter A Valid Code!
Item Code:
```

Figure 11 – Error Handling Figure 2

```
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : did
Enter Item Code: 3
This Item Code NOT Found In The System!
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC :
```

Figure 12 – Error Handling Figure 3

```
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : uid
Updating an item...
Enter item code: 4
Item NOT Found!
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC :
```

Figure 13 – Error Handling Figure 4

```
Item Price: d
Please Enter A Valid Price!
Item Price:
```

Figure 14 – Error Handling Figure 5

```
Item Quantity: d
Please Enter A Valid Quantity!
Item Quantity:
```

Figure 15 – Error Handling Figure 6

```
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : vrl
Please Select the dealer's first.
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC :
```

Figure 16 – Error Handling Figure 7

```
Enter your choice 'AID, DID, UID, VID, SID, SDD, VRL, LDI, ESC : lar
Enter dealer name 1 of 4: hj
Dealer not found. Please try again.
Enter dealer name 1 of 4:
```

Figure 17 – Error Handling Figure 8