INTRODUCTION

The **Computer and Accessories Sales Management System** is developed to efficiently manage customer information, product inventory, and sales records. The system, implemented in Python with MySQL integration, aims to streamline operations in a computer and accessories retail environment. This project involves creating and managing relational tables (Customer, Products, Sales) in MySQL, with operations facilitated through a Python interface.

OBJECTIVES

1. Database Connection and Setup:-

- Establishes a connection to the MySQL database with the sqltor.connect() function, ensuring all interactions are logged securely.
- Sets up the database tables (Customer, Products, Sales) with defined primary keys and foreign key relationships for efficient data storage and retrieval.

2. Customer Management:-

- insert_customer() : Adds new customers to the `Customer` table, capturing details like `Name`, `Contact`, and `Address`. This function also checks for duplicate contacts to avoid redundancy.
- update_customer(new_value, cont) : Updates existing customer details such as `Name`, `Contact`, or `Address` based on their contact number, ensuring data remains current.
- delete_customer(cont): Deletes a customer record using the contact number, supporting data maintenance and cleanup.
- view_customer(): Retrieves and displays all customer records in a tabular format, improving data accessibility and readability.
- check_customer_exists(cont): Verifies if a customer exists based on their contact number before a sale, allowing seamless addition of new customers when needed.

3. Product Management :

- insprod() : Adds new products to the `Products` table, including details like `Category`, `Name`, `Specifications`, `Stock`, and `Price`. This function helps keep track of all available products.
- delprod(pid) : Deletes a product based on its `P_ID` (Product ID), aiding in inventory management by removing discontinued items.
- Product Update Options : Provides a menu-driven interface to:
  - Update product name.
  - Update product specifications.
  - Update product stock by adding new stock to the existing value.
  - Update product price.
- view_products() : Displays all product records, presenting details in an organized table.
- category_view() : Displays products filtered by their category, allowing easy viewing of items within a specific group (e.g., Laptops, Desktops).
- search_by_product_name() : Searches products by name, displaying detailed information if found, or prompting for addition if not found.

4. Sales Management :

- check_product_stock(p_id, quantity) : Verifies if a sufficient quantity of a product is available before processing a sale, helping prevent stock inconsistencies.
- view_sales() : Displays all sales transactions with details like `Customer Name`, `Product ID`, `Quantity`, `Total Amount`, and more.
- Sales Processing :

    - Sale Entry : During a sale, the system checks for customer existence, verifies product availability, and calculates the total amount.
    - make_sale(): (incorporated in the main loop) Records the sale in the `Sales` table and updates the product stock, ensuring accuracy in both sales records and inventory.
- Customer Purchase History : Provides the option to view purchase history for a particular customer, enabling customer relationship management and personalized service.

5. Interactive Menu System :

- The main program runs a menu-driven interface for navigating between functions, making it intuitive for users to select and perform operations.
- Options include inserting, updating, and viewing records, managing stock, viewing sales, and exiting the system gracefully.

HARDWARE AND SOFTWARE REQUIREMENTS

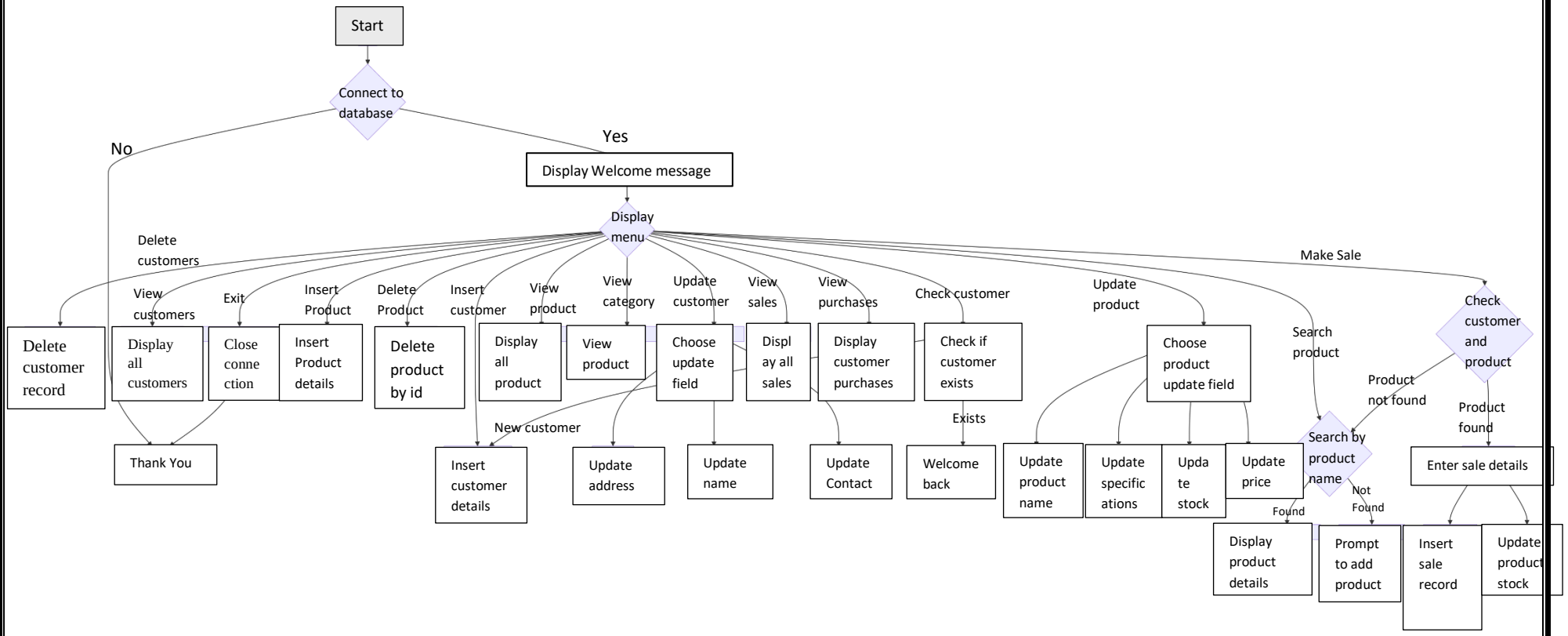
Hardware :-

1. Desktop Computer /Laptop
2. Mobile phone

Software :-

1. Python (latest version)
2. MySQL
3. MySQL-Connector-Python, Requests, Wikipedia-API, Datetime, Pyfiglet Modules

FLOW CHART


Start
Connect to database
No
Yes
Display Welcome message
Display menu
Delete customers
View customers
Exit
Insert Product
Delete Product
Insert customer
View product
View category
Update customer
View sales
View purchases
Check customer
Update product
Make Sale
Search product
Check customer and product
Delete customer record
Display all customers
Close connection
Insert Product details
Delete product by id
Display all product
View product
Choose update field
Displ ay all sales
Display customer purchases
Check if customer exists
Choose product update field
Product not found
Product found
New customer
Exists
Search by product name
Thank You
Insert customer details
Update address
Update name
Update Contact
Welcome back
Update product name
Update specific ations
Upda te stock
Update price
Enter sale details
Found
Not Found
Display product details
Prompt to add product
Insert sale record
Update product stock

PROJECT CODE

```
import mysql.connector as sqltor
# Establish connection to the database
con = sqltor.connect(
    host="localhost",
    user="group",
    password="G#n6Hg59",
    database="Computer_Sales_System")


if con.is_connected():
        print("||-------->>Connection successfully established<<--------||\n")
        print("<>-<>-<>-Welcome to Computer and Accessories Sales
        Management System-<>-<>-<>")
    cursor = con.cursor()
    # Creating table customers
    # cursor.execute("CREATE TABLE customer(c_id INT PRIMARY
KEY AUTO_INCREMENT, Name VARCHAR(40), Contact
VARCHAR(15), Address VARCHAR(255)) AUTO_INCREMENT =
10001")
```

```
    # print("Table CUSTOMER created successfully.")
    # Creating table products
    # cursor.execute("CREATE TABLE products (p_id INT PRIMARY KEY AUTO_INCREMENT, Category VARCHAR(100), Name VARCHAR(500), Specifications VARCHAR(1000), Stock INT, Price FLOAT) AUTO_INCREMENT = 51029")
    # print("Table PRODUCTS created successfully.")
    # Creating table Sales
    # cursor.execute("CREATE TABLE Sales (Customer_ID INT PRIMARY KEY, Customer_Name VARCHAR(255), Product_ID INT, Product_Name VARCHAR(255), Price_of_1 FLOAT, Quantity INT, Total_Amount FLOAT)")
    # print("Table SALES created successfully.")
    def insert_customer():  # Insert details of customer
        while True:
            name = input("\nEnter customer's name: ").title()
            contact = input("\nEnter customer's contact number: ")
            address = input("\nEnter customer's address: ")
            cursor.execute("INSERT INTO CUSTOMER (Name, Contact, Address) VALUES (%s, %s, %s)",(name, contact, address))
            con.commit()
```

```
        print("Details inserted successfully!")
        ch = input("Do you want to enter more records? (y/n): ")
        if ch in 'Nn':
            break


    def update_customer(new_value, id_value):  # Update customer details
        up = int(input("What do you want to update?\nEnter\n1 for updating customer's name\n2 for contact number\n3 for address\nchoice: "))
        if up == 1:
            query = "UPDATE CUSTOMER SET Name = %s WHERE C_ID = %s;"
        elif up == 2:
            query = "UPDATE CUSTOMER SET Contact = %s WHERE C_ID = %s;"
        elif up == 3:
            query = "UPDATE CUSTOMER SET Address = %s WHERE C_ID = %s;"
        else:
            print("Invalid option!")
            return
```

```
        cursor.execute(query, (new_value, id_value))
        con.commit()
        print("Details updated successfully!")


    def insprod():  # Insert data to table product
        while True:
            cat=input("\nEnter Product Category: ").title()
            pname = input("\nEnter Name of the Product: ")
            specs = input("\nEnter Specifications of the above product
(separated by commas): ").title()
            stock = int(input("\nHow many Stocks of the above product are
left?: "))
            price = float(input("\nEnter the Price of the above product: ₹
"))
            cursor.execute("INSERT INTO PRODUCTS (Category,
Name, Specifications, Stock, Price) VALUES (%s, %s, %s, %s, %s)",
                (cat, pname, specs, stock, price))
            con.commit()
            print("Data inserted successfully!")
            ch = input("Do you want to enter more records? (y/n): ")
```

```
            if ch in 'Nn':
                break

    def delprod(pid):  # Delete product record
        cursor.execute("DELETE FROM PRODUCTS WHERE P_ID =
%s", (pid,))
        con.commit()
        if cursor.rowcount > 0:
            print(f"Product with P_ID {pid} deleted successfully.")
        else:
            print(f"No product found with P_ID {pid}.")

    def delete_customer(id_value):  # Delete customer record
        query = "DELETE FROM CUSTOMER WHERE C_ID = %s"
        cursor.execute(query, (id_value,))
        con.commit()
        print("Record deleted successfully.")

    def view_customer():  # Display table customer
```

```
        query = "SELECT * FROM CUSTOMER"
        cursor.execute(query)
        results = cursor.fetchall()
        print("-----------------------------CUSTOMER-------------------------
----")
        print(f"{'C_ID':<10} {'Name':<40} {'Contact':<15}
{'Address':<40}")
        print()
        for row in results:
            print(f"{row[0]:<10} {row[1]:<40} {row[2]:<15}
{row[3]:<40}")


    def search_by_product_name():
        n = input("Enter name of the Product to be searched: ")
        cursor.execute("SELECT Name FROM PRODUCTS WHERE
Name = %s", (n,))
        result = cursor.fetchone()
        if result is not None:
            print("Found!\n")
            cursor.execute("SELECT * FROM PRODUCTS WHERE Name
= %s", (n,))
```

```
        results = cursor.fetchall()
        print(f"{'P_ID':<10} {'Name':<40} {'Specifications':<80} {'Stock':<10} {'Price (₹)':<10}")
        for row in results:
            print(f"{row[0]:<10} {row[1]:<40} {row[2]:<80} {row[3]:<10} {row[4]:<10}")
        return True
    else:
        print("\nSorry, NOT Found..")
        return False
def check_customer_exists(cname):
    try:
        query = "SELECT 1 FROM Customer WHERE Name = %s"
        cursor.execute(query, (cname,))
        result=cursor.fetchone()
        if result is None:
            print("New Customer?..Adding to Database.")
            insert_customer()
        else:
            print("Welcome Back!")
```

```
        except sqltor.Error as e:
            print(f"Database error: {e}")
            return False
    def view_products():  # Display table products
        cursor.execute("SELECT * FROM PRODUCTS")
        results = cursor.fetchall()
        print("------------------------------PRODUCTS-------------------------
---")
        print(f"{'P_ID':<10} {'Name':<40} {'Specifications':<80}
{'Stock':<10} {'Price (₹)':<10}")
        print()
        for row in results:
            print(f"{row[0]:<10} {row[1]:<40} {row[2]:<80} {row[3]:<10}
{row[4]:<10}")
    def category_view():
        c = input("Enter Category (Laptop/Desktop/Processor, etc.):
").title()
        cursor.execute("SELECT * FROM PRODUCTS WHERE Category
= %s", (c,))
        results = cursor.fetchall()  # Fetch all results
        if results:  # Check if results is not empty
```

```
        print(f"\n---------------------------- {c}s ----------------------------")
        print(f"{'P_ID':<10} {'Category':<30} {'Name':<25} {'Specifications':<70} {'Stock':<7} {'Price (₹)':<10}")
        print()
        for row in results:
            print(f"{row[0]:<10} {row[1]:<30} {row[2]:<25} {row[3]:<70} {row[4]:<7} {row[5]:<10}")
            return True  # Products were found, return True
    else:
        print(f"\nNo products found in the '{c}' category.")
        return False


def check_product_stock(p_id, quantity):
    try:
        query = "SELECT stock FROM products WHERE p_id = ? LIMIT 1;"
        cursor.execute(query, (p_id,))
        result = cursor.fetchone()
```

```
        if result is not None:
            available_stock = result[0]
            if available_stock >= quantity:
             return True
            else:
                print("Insufficient stock!")
                return False
        else:
            print("Product not found!")
            return False
    except sqltor.Error as e:
        print(f"Database error: {e}")
        return False
def view_sales(): # Display table SALES
    query = "SELECT * FROM SALES"
    cursor.execute(query)
    results = cursor.fetchall()
    print("-----------------------------SALES-----------------------------")
```

```
        print(f"{'Customer_ID':<10} {'Customer_NAME':<40} {'Product_ID':<10} {'Product_Name':<40} {'Price(1)':<10} {'Quantity':<15} {'Total_Amount(₹)':<10}")
        print()
        for row in results:
            print(f"{row[0]:<10} {row[1]:<40} {row[2]:<10} {row[3]:<40} {row[4]:<10} {row[5]:<15} {row[6]:<10}")


    i = 1
    while i != 12:
        print("\n1. Insert Data into table Customer.\n2. Delete Data from table Customer.")
        print("3. Update Data of table Customer.\n4. View Data of table Customer.")
        print("\n5. Insert Data into table Products\n6. Delete Data from table Products.")
        print("7. Update Data of table Products.\n8. View Data of table Products.\n9. View Products by Category.")
        print("\n10. Make Sale.\n11. View Sales.\n\n12. Exit")
        i = int(input("Enter choice (1/2/3/4/5/6/7/8/9/10/11/12): "))
        if i == 1:
```

```
        insert_customer()
    elif i == 2:
        id_value2 = int(input("Enter customer ID of customer whose details are to be deleted: "))
        delete_customer(id_value2)
    elif i == 3:
        id_value1 = int(input("Enter customer ID of customer whose details are to be updated: ")).title()
        new_value = input("Enter the value to be updated: ").title()
        update_customer(new_value, id_value1)
    elif i == 4:
        view_customer()
    elif i == 5:
        insprod()
    elif i == 6:
        p_id = int(input("\n\nEnter Product ID: "))
        delprod(p_id)
    elif i == 7:
        p_id = int(input("\n\nEnter Product ID: "))
```

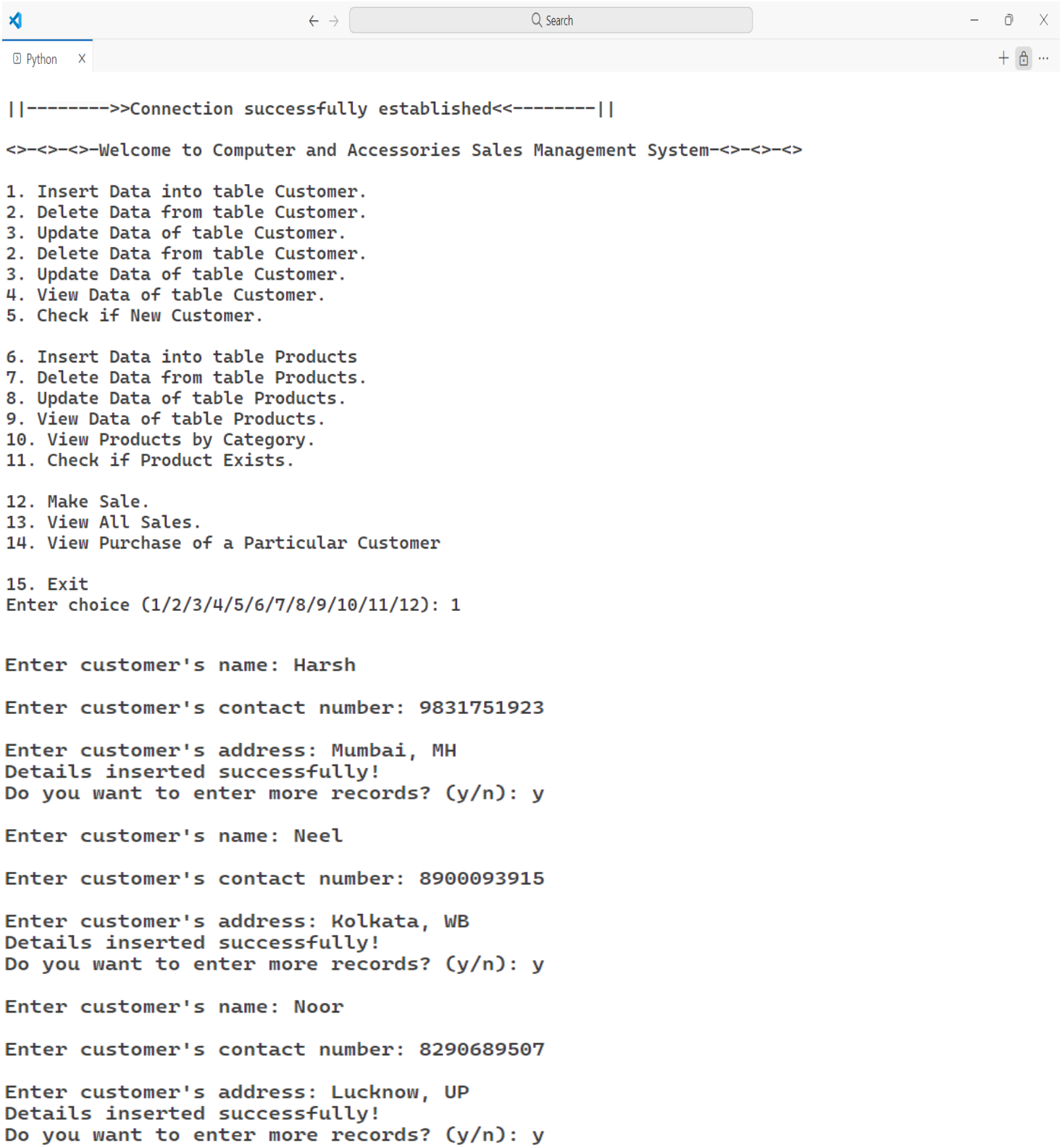
```
        print("Update\n1. Name of the Product\n2. Specifications of Product\n3. Stock of Product\n4. Price of Product")
        c = int(input("Enter choice (1/2/3/4): "))
        if c == 1:
            n = input("\nEnter new Name of the product: ").title()
            cursor.execute("UPDATE PRODUCTS SET Name = %s WHERE P_ID = %s", (n, p_id))
            con.commit()
            print("Name updated successfully!")
        elif c == 2:
            s = input("\nEnter new Specifications of the product: ")
            cursor.execute("UPDATE PRODUCTS SET Specifications = %s WHERE P_ID = %s", (s, p_id))
            con.commit()
            print("Specifications updated successfully!")
        elif c == 3:
            st = int(input("\nEnter new Stock number of the product: "))
            cursor.execute("UPDATE PRODUCTS SET Stock = %s WHERE P_ID = %s", (st, p_id))
            con.commit()
```

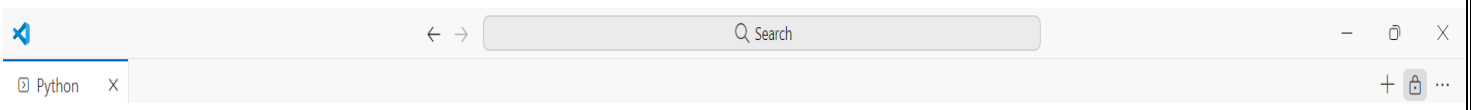
```
            print("Stock Number updated successfully!")
        elif c == 4:
            p = float(input("\nEnter new Price of the product: ₹ "))
            cursor.execute("UPDATE PRODUCTS SET Price = %s
WHERE P_ID = %s", (p, p_id))
            con.commit()
            print("Price updated successfully!")
    elif i == 8:
        view_products()
    elif i == 9:
        category_view()
    elif i == 10:


            cn = input("Enter Customer Name: ").title()
            check_customer_exists(cn)
            c12=int(input("1. Search Product by Name\n2. View Products
by Category\nChoice (1/2): "))
            chk=0
            if c12==1:
                if search_by_product_name()==True:
```

```
            chk=1
        elif c12==2:
            if category_view()==True:
                chk=1
        if chk==1:
            pid1=int(input("Enter Product ID (from table above): "))
            quan=int(input("Quantity: "))
            if check_product_stock(pid1,quan)==True:
                cursor.execute("SELECT c_id, name FROM Customer
WHERE name = %s", (cn,))
                cust = cursor.fetchone()
                cursor.execute("SELECT p_id, name, price FROM
Products WHERE p_id = %s", (pid1,))
                prod = cursor.fetchone()
                if cust and prod:
                    total_amt = quan * prod[2]
                    cursor.execute("INSERT INTO Sales (Customer_ID,
Customer_Name, Product_ID, Product_Name, Price_of_1, Quantity,
Total_Amount) VALUES (%s,%s,%s,%s,%s,%s,%s)", (cust[0], cust[1],
prod[0], prod[1],prod[2], quan, total_amt))
                    con.commit()
```

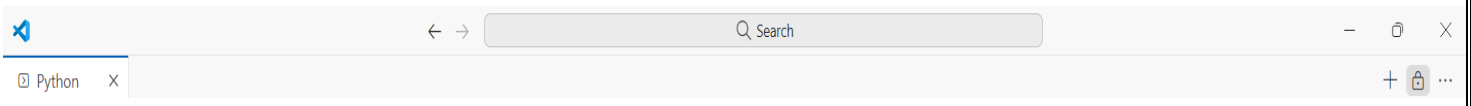
```
                    print("Data successfully inserted into the Sales table.")
                    cursor.execute("UPDATE Products SET Stock = Stock
- %s WHERE p_id = %s", (quan, prod[0]))
                    con.commit()
                else:
                    print("Customer or Product not found.")
            else:
              print("purchase can't be made...retry with lesser quantity!")
        else:
             print("Product Not Found!")
    elif i == 11:
        view_sales()
    elif i == 12:
        cursor.close()
        con.close()
        print("Exited successfully.\nTHANK YOU")
    input("\nPress Enter to proceed...")
else:
   print("CONNECTION UNSUCCESSFUL!")
```

SCREENSHOTS OF OUTPUT

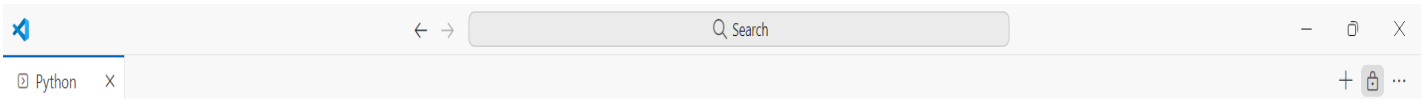




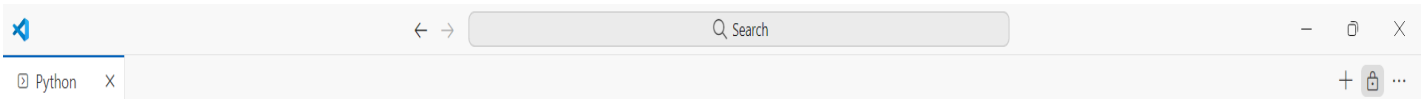

```
Enter customer's name: Mohan

Enter customer's contact number: 4288019808

Enter customer's address: Ahmedabad, GJ
Details inserted successfully!
Do you want to enter more records? (y/n): n

Press Enter to proceed...

1. Insert Data into table Customer.
2. Delete Data from table Customer.
3. Update Data of table Customer.
4. View Data of table Customer.
5. Check if New Customer.

6. Insert Data into table Products
7. Delete Data from table Products.
8. Update Data of table Products.
9. View Data of table Products.
10. View Products by Category.
11. Check if Product Exists.

12. Make Sale.
13. View All Sales.
14. View Purchase of a Particular Customer

15. Exit
Enter choice (1/2/3/4/5/6/7/8/9/10/11/12): 4
------------------------------CUSTOMER-------------------------------
Name                                    Contact         Address

Mohan                                   4288019808      Ahmedabad, GJ
Noor                                    8290689507      Lucknow, UP
Neel                                    8900093915      Kolkata, WB
Harsh                                   9831751923      Mumbai, MH

Press Enter to proceed...
```



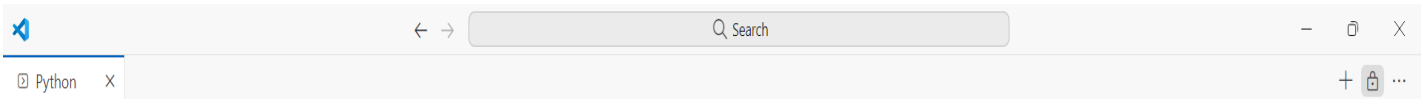

```
1. Insert Data into table Customer.
2. Delete Data from table Customer.
3. Update Data of table Customer.
4. View Data of table Customer.
5. Check if New Customer.

6. Insert Data into table Products
7. Delete Data from table Products.
8. Update Data of table Products.
9. View Data of table Products.
10. View Products by Category.
11. Check if Product Exists.

12. Make Sale.
13. View All Sales.
14. View Purchase of a Particular Customer

15. Exit
Enter choice (1/2/3/4/5/6/7/8/9/10/11/12): 3
Enter Contact of customer whose details are to be updated: 8900093915
Enter the value to be updated: Howrah, WB
What do you want to update?
Enter
1 for updating customer's name
2 for contact number
3 for address
choice: 3
Details updated successfully!




Press Enter to proceed...

1. Insert Data into table Customer.
2. Delete Data from table Customer.
3. Update Data of table Customer.
4. View Data of table Customer.
5. Check if New Customer.

6. Insert Data into table Products
7. Delete Data from table Products.
8. Update Data of table Products.
9. View Data of table Products.
10. View Products by Category.
11. Check if Product Exists.

12. Make Sale.
13. View All Sales.
14. View Purchase of a Particular Customer

15. Exit
Enter choice (1/2/3/4/5/6/7/8/9/10/11/12): 2
Enter Contact of customer whose details are to be deleted: 4288019808
Record deleted successfully.
```



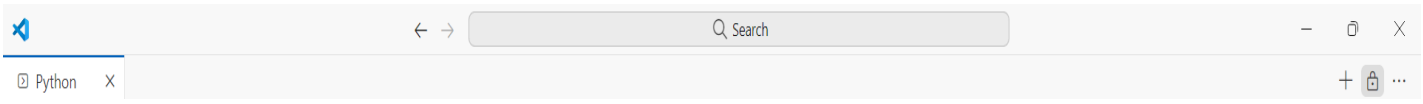

```
Press Enter to proceed...

1. Insert Data into table Customer.
2. Delete Data from table Customer.
3. Update Data of table Customer.
4. View Data of table Customer.
5. Check if New Customer.

6. Insert Data into table Products
7. Delete Data from table Products.
8. Update Data of table Products.
9. View Data of table Products.
10. View Products by Category.
11. Check if Product Exists.

12. Make Sale.
13. View All Sales.
14. View Purchase of a Particular Customer

15. Exit
Enter choice (1/2/3/4/5/6/7/8/9/10/11/12): 4
------------------------------CUSTOMER------------------------------
Name                                    Contact         Address

Noor                                    8290689507      Lucknow, UP
Neel                                    8900093915      Howrah, Wb
Harsh                                   9831751923      Mumbai, MH

Press Enter to proceed...
```



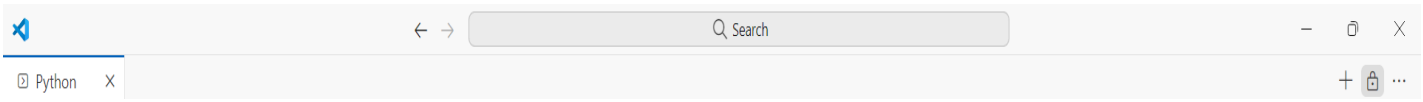

```
1. Insert Data into table Customer.
2. Delete Data from table Customer.
3. Update Data of table Customer.
4. View Data of table Customer.
5. Check if New Customer.

6. Insert Data into table Products
7. Delete Data from table Products.
8. Update Data of table Products.
9. View Data of table Products.
10. View Products by Category.
11. Check if Product Exists.

12. Make Sale.
13. View All Sales.
14. View Purchase of a Particular Customer

15. Exit
Enter choice (1/2/3/4/5/6/7/8/9/10/11/12): 6

Enter Product Category: Mouse

Enter Name of the Product: HP M160

Enter Specifications of the above product (separated by commas): RGB Gaming Mouse, 300 DPI

How many Stocks of the above product are left?: 5

Enter the Price of the above product: ₹ 300
Data inserted successfully!
Do you want to enter more records? (y/n): y

Enter Product Category: Keyboard

Enter Name of the Product: Logitech K380

Enter Specifications of the above product (separated by commas): Bluetooth Keyboard, 3 devices connectivity

How many Stocks of the above product are left?: 3

Enter the Price of the above product: ₹ 2800
Data inserted successfully!
Do you want to enter more records? (y/n): y
```




```
Enter Product Category: Graphics Card

Enter Name of the Product: NVIDIA RTX 4080

Enter Specifications of the above product (separated by commas): 16 GB VRAM, GDDR6X

How many Stocks of the above product are left?: 2

Enter the Price of the above product: ₹ 110000
Data inserted successfully!
Do you want to enter more records? (y/n): y

Enter Product Category: Processor

Enter Name of the Product: AMD Ryzen 7 5700X3D Desktop Processor

Enter Specifications of the above product (separated by commas): 8 cores 16 Threads 3 GHz-4.1 Ghz

How many Stocks of the above product are left?: 5

Enter the Price of the above product: ₹ 22390
Data inserted successfully!
Do you want to enter more records? (y/n): n

Press Enter to proceed...
```



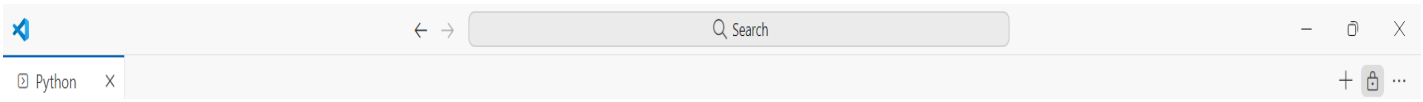

```
1. Insert Data into table Customer.
2. Delete Data from table Customer.
3. Update Data of table Customer.
2. Delete Data from table Customer.
3. Update Data of table Customer.
4. View Data of table Customer.
5. Check if New Customer.

6. Insert Data into table Products
7. Delete Data from table Products.
8. Update Data of table Products.
9. View Data of table Products.
10. View Products by Category.
11. Check if Product Exists.

12. Make Sale.
13. View All Sales.
14. View Purchase of a Particular Customer

15. Exit
Enter choice (1/2/3/4/5/6/7/8/9/10/11/12): 8


Enter Product ID: 51032
Update
1. Name of the Product
2. Specifications of Product
3. Stock of Product
4. Price of Product
Enter choice (1/2/3/4): 1

Enter new Name of the product: AMD Ryzen 7 5700X3D
Name updated successfully!

Press Enter to proceed...
```




```
1. Insert Data into table Customer.
2. Delete Data from table Customer.
3. Update Data of table Customer.
4. View Data of table Customer.
5. Check if New Customer.

6. Insert Data into table Products
7. Delete Data from table Products.
8. Update Data of table Products.
9. View Data of table Products.
10. View Products by Category.
11. Check if Product Exists.

12. Make Sale.
13. View All Sales.
14. View Purchase of a Particular Customer

15. Exit
Enter choice (1/2/3/4/5/6/7/8/9/10/11/12): 9
```

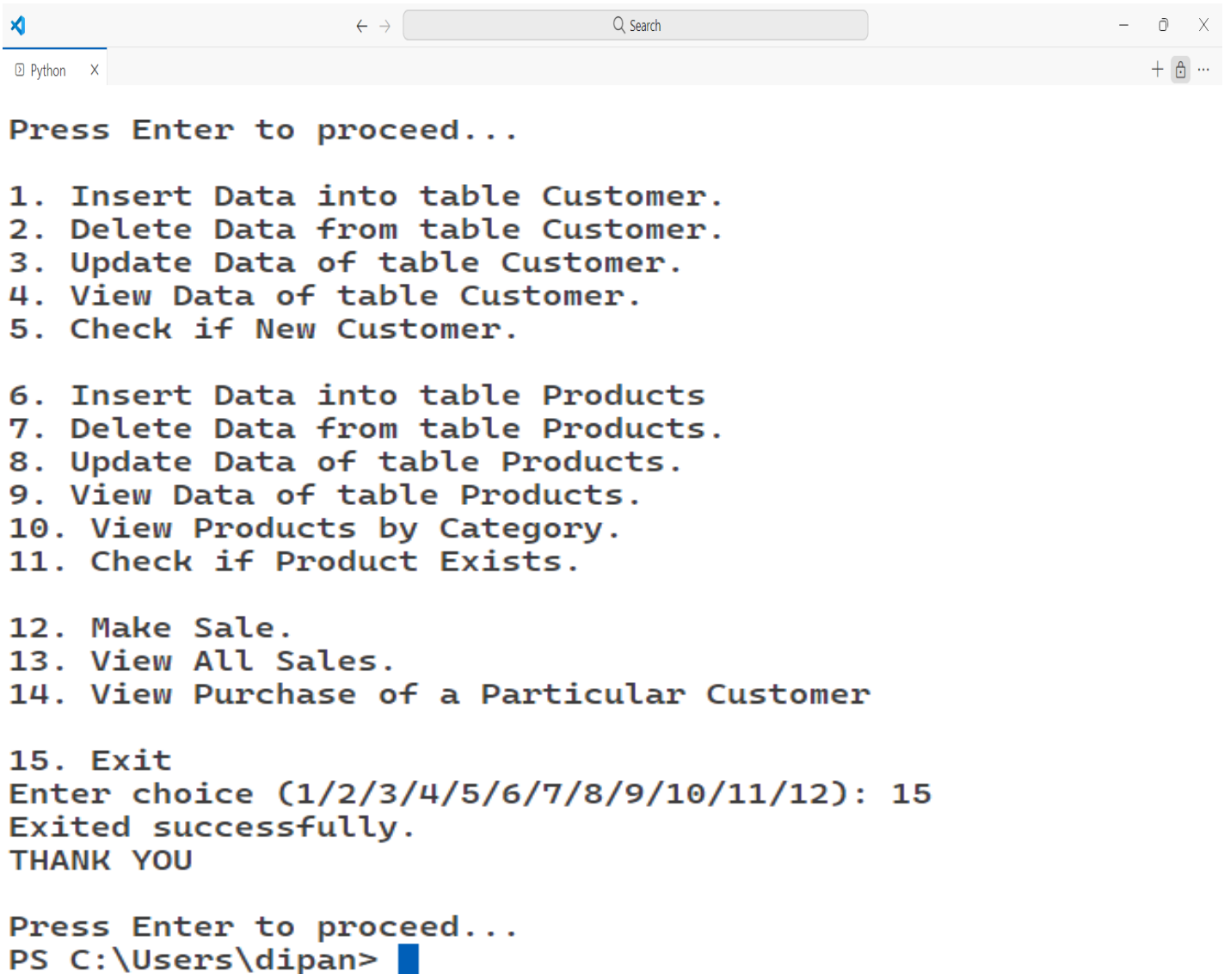
----------PRODUCTS----------

P_ID	Category	Name	Specifications	Stock	Price (₹)
51029	Mouse	HP M160	Rgb Gaming Mouse, 300 Dpi	5	300.0
51030	Keyboard	Logitech K380	Bluetooth Keyboard, 3 Devices Connectivity	3	2800.0
51031	Graphics Card	NVIDIA RTX 4080	16 Gb Vram, Gddr6X	2	110000.0
51032	Processor	Amd Ryzen 7 5700X3D	8 Cores 16 Threads 3 Ghz-4.1 Ghz	5	22390.0




```
Press Enter to proceed...

1. Insert Data into table Customer.
2. Delete Data from table Customer.
3. Update Data of table Customer.
4. View Data of table Customer.
5. Check if New Customer.

6. Insert Data into table Products
7. Delete Data from table Products.
8. Update Data of table Products.
9. View Data of table Products.
10. View Products by Category.
11. Check if Product Exists.

12. Make Sale.
13. View All Sales.
14. View Purchase of a Particular Customer

15. Exit
Enter choice (1/2/3/4/5/6/7/8/9/10/11/12): 12
Enter Customer Phone Number: 8900093915
Welcome Back!
1. Search Product by Name
2. View Products by Category
Choice (1/2): 2
Enter Category (Laptop/Desktop/Processor, etc.): Mouse

------------------------------ Mouses ------------------------------
P_ID      Category                  Name                  Specifications                                                  Stock   Price (₹)

51029     Mouse                     HP M160               Rgb Gaming Mouse, 300 Dpi                                       5       300.0
Enter Product ID (from table above): 51029
Quantity: 2
Data successfully inserted into the Sales table.
Stock for product ID 51029 updated successfully.


Press Enter to proceed...

1. Insert Data into table Customer.
2. Delete Data from table Customer.
3. Update Data of table Customer.
4. View Data of table Customer.
5. Check if New Customer.

6. Insert Data into table Products
7. Delete Data from table Products.
8. Update Data of table Products.
9. View Data of table Products.
10. View Products by Category.
11. Check if Product Exists.

12. Make Sale.
13. View All Sales.
14. View Purchase of a Particular Customer

15. Exit
Enter choice (1/2/3/4/5/6/7/8/9/10/11/12): 12
```

```
Enter Customer Phone Number: 8434329623
New Customer..Want to add to database? (Y/N)y

Enter customer's name: Stuti

Enter customer's contact number: 8434329623

Enter customer's address:  Kishtwar, J&K
Details inserted successfully!
Do you want to enter more records? (y/n): n
1. Search Product by Name
2. View Products by Category
Choice (1/2): 1
Enter name of the Product to be searched: AMD Ryzen 7 5700X3D
Found!


------------------------------ AMD Ryzen 7 5700X3D ------------------------------
P_ID      Category                  Name                  Specifications                                          Stock   Price (₹)

51032     Processor                 Amd Ryzen 7 5700X3D   8 Cores 16 Threads 3 Ghz-4.1 Ghz                        5
                                   22390.0
Enter Product ID (from table above): 51032
Quantity: 1
Data successfully inserted into the Sales table.
Stock for product ID 51032 updated successfully.

Press Enter to proceed...


1. Insert Data into table Customer.
2. Delete Data from table Customer.
3. Update Data of table Customer.
4. View Data of table Customer.
5. Check if New Customer.

6. Insert Data into table Products
7. Delete Data from table Products.
8. Update Data of table Products.
9. View Data of table Products.
10. View Products by Category.
11. Check if Product Exists.

12. Make Sale.
13. View All Sales.
14. View Purchase of a Particular Customer

15. Exit
Enter choice (1/2/3/4/5/6/7/8/9/10/11/12): 13
------------------------------SALES------------------------------
Customer_NAME             Contact        Product_ID Product_Name                Price_of_1 Quantity       Total_Amount(₹)

Neel                      8900093915     51029      HP M160                    300.0      2              600.0
Stuti                     8434329623     51032      Amd Ryzen 7 5700X3D        22390.0    1              22390.0
```


Search
Python
Press Enter to proceed...
1. Insert Data into table Customer.
2. Delete Data from table Customer.
3. Update Data of table Customer.
4. View Data of table Customer.
5. Check if New Customer.
6. Insert Data into table Products
7. Delete Data from table Products.
8. Update Data of table Products.
9. View Data of table Products.
10. View Products by Category.
11. Check if Product Exists.
12. Make Sale.
13. View All Sales.
14. View Purchase of a Particular Customer
15. Exit
Enter choice (1/2/3/4/5/6/7/8/9/10/11/12): 15
Exited successfully.
THANK YOU
Press Enter to proceed...
PS C:\Users\dipan>

SCREENSHOTS OF DATABASE

1. Table Customer

Name	Contact	Address
Noor	8290689507	Lucknow, UP
Stuti	8434329623	Kishtwar, J&K
Neel	8900093915	Howrah, Wb
Harsh	9831751923	Mumbai, MH

2. Table Products

p_id	Category	Name	Specifications	Stock	Price
51029	Mouse	HP M160	Rgb Gaming Mouse, 300 Dpi	3	300
51030	Keyboard	Logitech K380	Bluetooth Keyboard, 3 Devices Connectivity	3	2800
51031	Graphics Card	NVIDIA RTX 4080	16 Gb Vram, Gddr6X	2	110000
51032	Processor	Amd Ryzen 7 5700X3D	8 Cores 16 Threads 3 Ghz-4.1 Ghz	4	22390

3. Table Sales

Customer_Name	Contact	Product_ID	Product_Name	Price_of_one	Quantity	Total_Amount
Neel	8900093915	51029	HP M160	300	2	600
Stuti	8434329623	51032	Amd Ryzen 7 5700X3D	22390	1	22390

LIMITATIONS

1. *Scalability*: The system may not be designed to handle a high volume of concurrent users or large datasets effectively.

2. *Security*: Basic security features such as user authentication and encrypted connections might be missing, posing risks of unauthorized access.

3. *User Interface*: The project may only include a simple user interface, focusing primarily on backend operations.

4. *Error Handling*: Limited error handling and data validation could lead to data integrity issues if improper input is entered.

5. *Advanced Features*: The system may not support advanced features like automated stock management or real-time updates.

6. *Reporting Capabilities*: The project may only include basic sales viewing features without comprehensive reporting or analytics tools

7. *Backup and Recovery*: A robust backup and recovery mechanism may be absent, risking data loss in case of failures.

BIBLIOGRAPHY

1. https://www.python.org/

2. *MySQL Documentation:* The MySQL documentation was referenced for understanding MySQL query syntax, creating tables, using primary keys and auto-increment properties in SQL. Available at: https://www.mysql.com/

3. *Python MySQL Connector Library Documentation:* This documentation provided details on using the mysql.connector library to connect Python applications with MySQL databases, as well as using functions like cursor.execute and con.commit. Available at: https://dev.mysql.com/doc/connector-python/en/

4. *Python Programming Textbook:* Concepts such as error handling with try-except, user input handling, and function definitions were developed using foundational knowledge from Python programming textbooks, such Computer Science Textbook for Class XII by NCERT. https://ncert.nic.in/