PROJECT DESCRIPTION

The Warehouse Management System (WMS) is a sophisticated software solution engineered to revolutionize the way in which warehouses operate, optimizing processes and bolstering efficiency. At its core, the WMS automates and streamlines inventory management, order processing, and reporting tasks, aiming to propel warehouse operations into the realm of maximum productivity and seamless workflow orchestration.

Utilizing a suite of cutting-edge tools and technologies, the WMS empowers warehouse managers and staff with a comprehensive toolkit to manage every aspect of their operations with precision and finesse. From Python to SQLite every component of the WMS is meticulously crafted to deliver unparalleled performance and reliability. Whether it's registering new warehouses, managing suppliers, processing orders, or generating insightful reports, users are guided through each step with clarity and simplicity, enabling swift decision-making and agile response to changing demands.

With Python scripts orchestrating the logic behind each operation and SQLite databases serving as the repository for critical warehouse data, the WMS ensures data integrity, reliability, and security at every turn. The output of the Warehouse Management System transcends mere data points and statistics, culminating in a symphony of actionable insights and informed decisions.

TOOLS USED

In the Warehouse Management System project, the following tools were utilized:

Programming Language:

Python: The project was primarily developed using Python, a versatile and powerful programming language known for its simplicity and readability.

Modules:

SQLite3: SQLite3 module was used for interacting with the SQLite database management system, enabling efficient storage and retrieval of data within the application.

DBeaver: DBeaver was used to create Entity-Relationship Diagram (ERD) for the database created.

Tabulate: The Tabulate module was employed to format query results into visually appealing tables, enhancing the readability of output data.

OS: The OS module facilitated interaction with the operating system, allowing for tasks such as file handling and process management within the application.

Other Standard Python Modules: Various standard Python modules were utilized for tasks such as user input handling, date-time operations and random number generation.

Software:

SQLite Database Browser: SQLite Database Browser software was used for viewing and managing the SQLite database files associated with the project.

Integrated Development Environment (IDE): An IDE such as Visual Studio Code was employed for writing, testing and debugging the Python code, providing features such as syntax highlighting, code completion, and debugging tools.

WORKING

1. Creating the SQLite Database File:

The code begins by importing necessary libraries: `sqlite3` for SQLite database operations, tabulate for formatting query results, and `os` for handling file paths. It establishes a connection to the SQLite database file named test.db located at `C:\\Users\\[username]\\Desktop\\Sqlite_project\\`.

2. Defining Menu Functions:

Each menu function corresponds to a specific aspect of the warehouse management system.

For instance,

Warehouse Menu:

Option 1: Registers a new warehouse

Option 2: Searches for warehouse by city

Option 3: Updates warehouse details

3. Database Interaction:

- The code interacts with the SQLite database using SQL queries executed through the sqlite3 library.
- SQL queries such as `INSERT`, `SELECT` and `UPDATE` statements are utilized to manipulate data in the database tables.
- Dynamic parameter passing ensures flexibility and security in database operations, accommodating user inputs seamlessly.

4. User Input and Validation:

- User input is acquired using the `input()` function, enabling interaction through the command-line interface (CLI).

- Input validation mechanisms verify the correctness and integrity of user-provided data, displaying error messages for invalid inputs.

5. Data Retrieval and Manipulation:

Total Price Calculation:

- Retrieves unit prices of products from the database.
- Calculates subtotal for each product by multiplying unit price with quantity and current tax rate.
- Addition of all subtotals to obtain the total price of all purchased products.

Quantity Reduction:

- Updates inventory records in the database to reflect reduced quantity of purchased items.
- Retrieves the available quantity of each product and subtracts purchased quantity.
- Stores updated quantity values back into the database.

6. Menu Navigation:

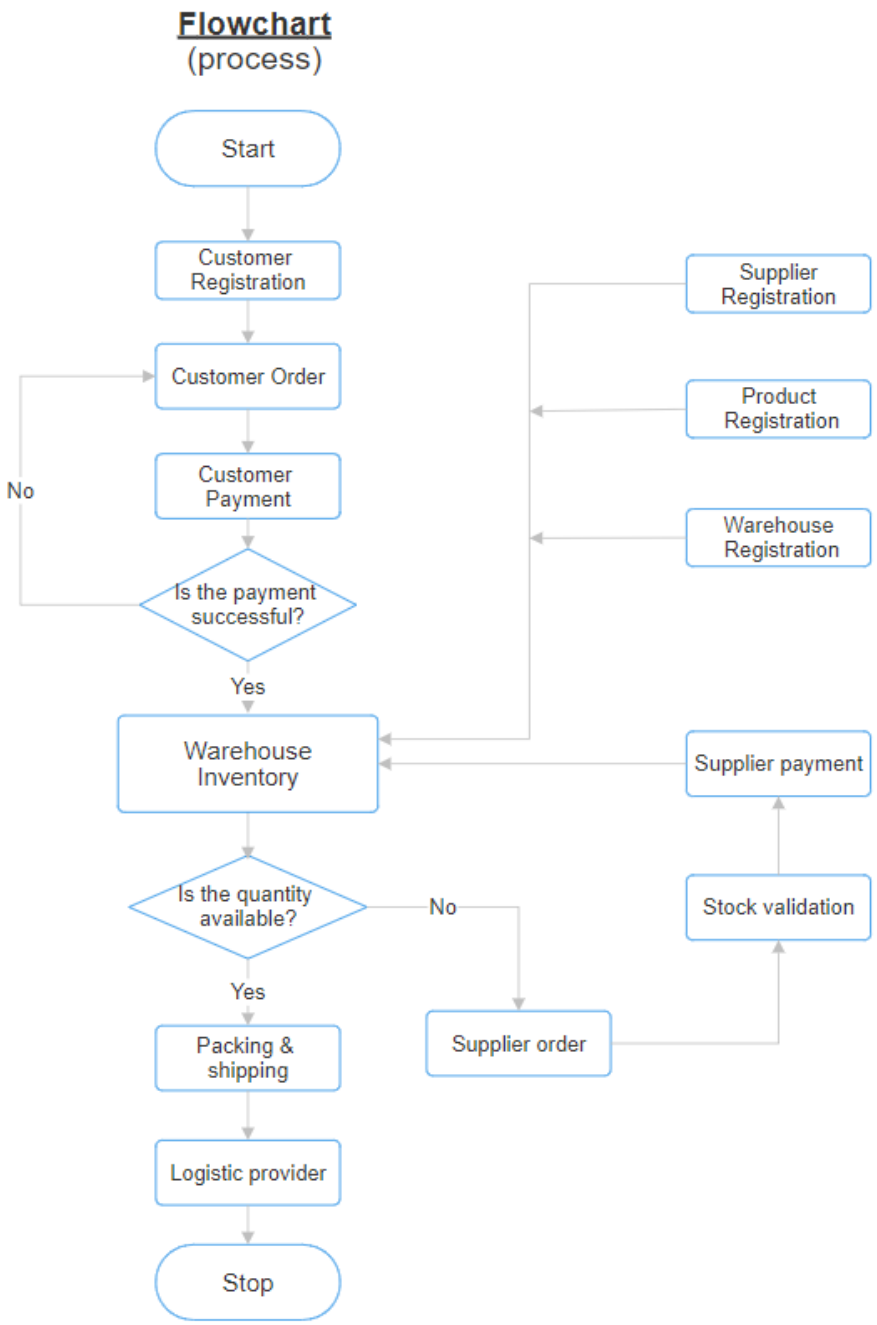
- Users are presented with options to return to the main menu or exit the program after completing a functionality.
- The main menu serves as a central hub for accessing diverse functionalities, enhancing navigation and user experience.

7. Main Menu Loop:

- The code operates within a perpetual loop, allowing users to navigate through various menu options iteratively.
- This loop structure ensures continuous program execution until users choose to exit, fostering uninterrupted interaction and task completion.

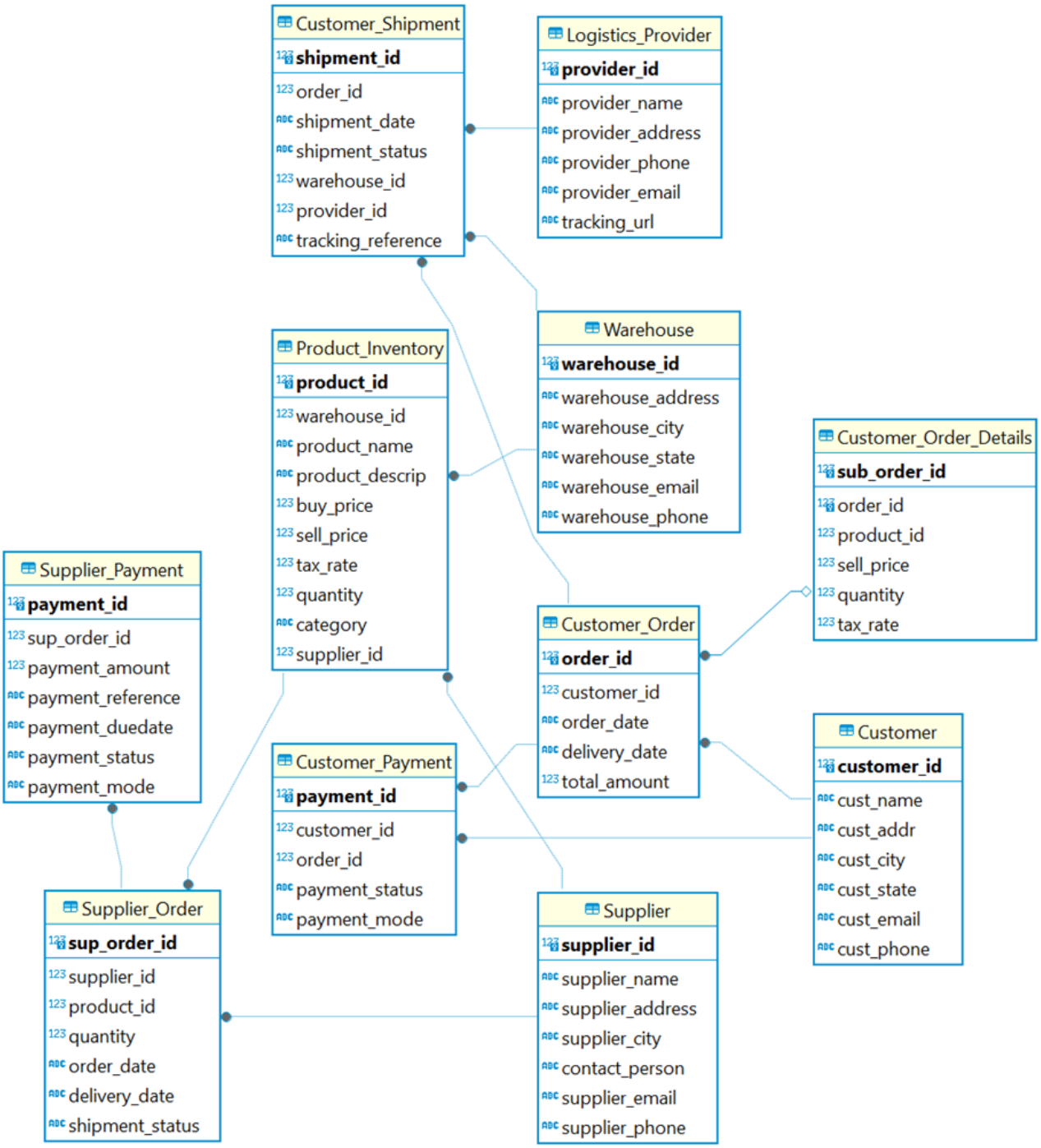
Process flowchart:

A process flowchart outlining the workflow and processes involved in the Warehouse Management System (WMS) is included below. This flowchart provides a graphical representation of the system's functionalities and the sequence of actions performed during its operation.



Entity-Relationship (ER) diagram:

An Entity-Relationship (ER) diagram illustrating the database schema used in the Warehouse Management System (WMS) is provided below. This diagram visually represents the entities, attributes and relationships within the database model.



SOURCE CODE

Python program: (warehouse.py)

```
import sqlite3
from tabulate import tabulate
import os
conn =
 sqlite3.connect('C:\\Users\\haris\\Desktop\\Sqlite_project\\warehouse\\wareho
 use.db')

# Menu for sqlite3 Warehouse Management project
def Warehouse_menu():
    while True:
        print('''
              --------------------------
              Warehouse menu:
              --------------------------
              1. Warehouse Registration
              2. Warehouse Search
              3. Warehouse Update
              0. Previous menu
              ''')
        W_option = int(input("Enter your option: "))
        if W_option == 1:
            w_address = input("Enter Warehouse address: ")
            w_city = input("Enter location of warehouse city: ")
            w_state = input("Enter the state of warehouse: ")
            w_email = input("Enter warehouse's e-mail address: ")
            w_phone = input("Enter warehouse's phone number: ")
            print(f"Warehouse Address: {w_address}\nWarehouse City:
 {w_city}\nWarehouse State: {w_state}\nWarehouse Email: {w_email}\nWarehouse
 Phone: {w_phone}")
            c = conn.cursor()
            c.execute('''INSERT INTO Warehouse(warehouse_address,
 warehouse_city, warehouse_state, warehouse_email, warehouse_phone) VALUES (?,
 ?, ?, ?, ?)''',(w_address, w_city, w_state, w_email, w_phone))
            conn.commit()
```

```
        elif W_option == 2:
            WCity = input("Enter the warehouse city: ")
            WCity = WCity.upper()
            print('\n')
            c = conn.cursor()
            c.execute("SELECT warehouse_id, warehouse_address, warehouse_city,
warehouse_state, warehouse_email, warehouse_phone FROM Warehouse WHERE
upper(warehouse_city)= ?",(WCity,))
            results =tabulate(c.fetchall(),headers=["warehouse_id",
"warehouse_address", "warehouse_city", "warehouse_state", "warehouse_email",
"warehouse_phone"])
            if results:
                print(results)
                print('\n')
            conn.commit()
        elif W_option == 3:
            w_identity = input("Enter Warehouse id: ")
            w_address = input("Enter Warehouse address: ")
            w_city = input("Enter location of warehouse city: ")
            w_state = input("Enter the state of warehouse: ")
            w_email = input("Enter warehouse's e-mail address: ")
            w_phone = input("Enter warehouse's phone number: ")
            c = conn.cursor()
            c.execute('''UPDATE Warehouse SET warehouse_address=?,
warehouse_city=?, warehouse_state=?, warehouse_email=?, warehouse_phone=?
WHERE Warehouse_id =?''', (w_address,w_city, w_state, w_email,
w_phone,w_identity))
            conn.commit()
        elif W_option == 0:
            MainMenu()
        else:
            print("Invalid Option")

def Supplier_menu():
    while True:
        print('''
              --------------------------
              Supplier menu:
              --------------------------
              1. Supplier Registration
              2. Supplier Search
```

```
            3. Supplier Update
            0. Previous menu
            ''')
        S_option = int(input("Enter your option: "))
        if S_option == 1:
            s_name = input("Enter supplier's name: ")
            s_contact_person = input("Enter contact person name for supplier:
")
            s_email = input("Enter supplier's e-mail address: ")
            s_phone = input("Enter supplier's phone number: ")
            s_address = input("Enter supplier's address: ")
            s_city = input("Enter supplier's city: ")
            print(f"Supplier Name: {s_name}\nContact Person:
{s_contact_person}\nSupplier Email: {s_email}\nSupplier Phone:
{s_phone}\nSupplier Address: {s_address}\nSupplier City: {s_city}")
            c = conn.cursor()
            c.execute('''INSERT INTO Supplier(supplier_name, contact_person,
supplier_email, supplier_phone, supplier_address, supplier_city) VALUES (?,
?, ?, ?, ?, ?)''',(s_name, s_contact_person, s_email, s_phone, s_address,
s_city))
            conn.commit()
        elif S_option == 2:
            SName = input("Enter supplier's name: ")
            SName = SName.upper()
            print('\n')
            c = conn.cursor()
            c.execute("SELECT supplier_id, supplier_name, supplier_address,
supplier_city, contact_person, supplier_email, supplier_phone FROM Supplier
WHERE upper(supplier_name)= ?",(SName,))
            results =tabulate(c.fetchall(),headers=["supplier_id",
"supplier_name", "supplier_address", "supplier_city", "contact_person",
"supplier_email", "supplier_phone"])
            if results:
                print(results)
                print('\n')
            conn.commit()
        elif S_option == 3:
            s_identity = input("Enter Supplier id: ")
            s_address = input("Enter Supplier address: ")
            s_city = input("Enter Supplier city: ")
            s_email = input("Enter Supplier's e-mail address: ")
```

```
            s_phone = input("Enter Supplier's phone number: ")
            c = conn.cursor()
            c.execute('''UPDATE Supplier SET supplier_address=?,
 supplier_city=?, supplier_email=?, supplier_phone=? WHERE supplier_id =?''',
 (s_address,s_city, s_email, s_phone, s_identity))
            conn.commit()
        elif S_option == 0:
            MainMenu()
        else:
            print("Invalid Option")

def Logistics_menu():
    while True:
        print('''
              --------------------------
              Logistics menu:
              --------------------------
              1. Logistics Registration
              2. Logistics Search
              3. Logistics Update
              0. Previous menu
              ''')
        L_option = int(input("Enter your option: "))
        if L_option == 1:
            L_name = input("Enter Logistics provider's name: ")
            L_address = input("Enter Logistics provider's address: ")
            L_phone = input("Enter Logistics provider's phone number: ")
            L_email = input("Enter Logistics provider's e-mail address: ")
            L_track_url = input("Enter Logistics provider's tracking url: ")
            print(f"Logistics provider's name: {L_name}\nLogistics provider's
 address: {L_address}\nLogistics provider's phone: {L_phone}\nLogistics
 provider's email: {L_email}\nLogistics provider's tracking url:
 {L_track_url}")
            c = conn.cursor()
            c.execute('''INSERT INTO Logistics_Provider(provider_name,
 provider_address, provider_phone, provider_email, tracking_url) VALUES (?, ?,
 ?, ?, ?)''',(L_name, L_address, L_phone, L_email, L_track_url))
            conn.commit()
        elif L_option == 2:
            LName = input("Enter Logistics provider's name: ")
            LName = LName.upper()
```

```
            print('\n')
            c = conn.cursor()
            c.execute("SELECT provider_id, provider_name, provider_address,
provider_phone, provider_email, tracking_url FROM Logistics_Provider WHERE
upper(provider_name)= ?",(LName,))
            results =tabulate(c.fetchall(),headers=["provider_id",
"provider_name", "provider_address", "provider_phone", "provider_email",
"tracking_url"])
            if results:
                print(results)
                print('\n')
            conn.commit()
        elif L_option == 3:
            L_identity = input("Enter Logistics provider's id: ")
            L_address = input("Enter Logistics provider's address: ")
            L_email = input("Enter Logistics provider's e-mail address: ")
            L_phone = input("Enter Logistics provider's phone number: ")
            c = conn.cursor()
            c.execute('''UPDATE Logistics_Provider SET provider_address=?,
provider_email=?, provider_phone=? WHERE provider_id =?''', (L_address,
L_email, L_phone, L_identity))
            conn.commit()
        elif L_option == 0:
            MainMenu()
        else:
            print("Invalid Option")

def Product_menu():
    while True:
        print('''
              --------------------------
              Product menu:
              --------------------------
              1. Product Registration
              2. Product Search
              3. Product Update
              0. Previous menu
              ''')
        P_option = int(input("Enter your option: "))
        if P_option == 1:
            P_name = input("Enter product's name: ")
```

```
            P_descrip = input("Enter product's description: ")
            P_buy_price = float(input("Enter product's buying price: "))
            P_sell_price = float(P_buy_price*1.25)
            P_taxrate = float(input("Enter product's tax rate: "))
            P_quantity = int(input("Enter quantity of product: "))
            P_category = input("Enter category of product: ")
            P_sup_id = int(input("Enter supplier id: "))
            P_warehouse_id = int(input("Enter warehouse id: "))
            print(f"Product's name: {P_name}\nProduct's description:
{P_descrip}\nProduct's buying price: {P_buy_price}\nProduct's selling price:
{P_sell_price}\nProduct's tax rate: {P_taxrate}\nQuantity of product:
{P_quantity}\nCategory of product: {P_category}\nSupplier id:
{P_sup_id}\nWarehouse id: {P_warehouse_id}")
            c = conn.cursor()
            c.execute('''INSERT INTO Product_Inventory(product_name,
product_descrip, buy_price, sell_price, tax_rate, quantity, category,
supplier_id, warehouse_id) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)''',(P_name,
P_descrip, P_buy_price, P_sell_price, P_taxrate, P_quantity, P_category,
P_sup_id, P_warehouse_id))
            conn.commit()
        elif P_option == 2:
            PCategory = input("Enter product's category: ")
            PCategory = PCategory.upper()
            print('\n')
            c = conn.cursor()
            c.execute("SELECT product_id, product_name, product_descrip,
buy_price, sell_price, tax_rate, quantity, category, supplier_id,
warehouse_id FROM Product_Inventory WHERE upper(category)= ?",(PCategory,))
            results =tabulate(c.fetchall(),headers=["prod_id", "prod_name",
"prod_descrip", "buy_price", "sell_price", "tax_rate", "quantity",
"category", "supplier_id", "warehouse_id"])
            if results:
                print(results)
                print('\n')
            conn.commit()
        elif P_option == 3:
            P_identity = input("Enter Product's id: ")
            P_name = input("Enter updated product name: ")
            P_prod_descrip = input("Enter new description: ")
            P_buy_price = float(input("Enter product's buying price: "))
            P_sell_price = float(P_buy_price*1.25)
```

```
            P_taxrate = float(input("Enter product's tax rate: "))
            P_quantity = int(input("Enter updated quantity of product: "))
            P_category = input("Enter updated category: ")
            P_sup_id = int(input("Enter updated supplier id: "))
            P_warehouse_id = int(input("Enter updated warehouse id: "))
            c = conn.cursor()
            c.execute('''UPDATE Product_Inventory SET warehouse_id=?,
 product_name=?, product_descrip=?, buy_price=?, sell_price=?, tax_rate=?,
 quantity=?, category=?, supplier_id=? WHERE product_id =?''',
 (P_warehouse_id, P_name, P_prod_descrip, P_buy_price, P_sell_price,
 P_taxrate, P_quantity, P_category, P_sup_id, P_identity))
            conn.commit()
        elif P_option == 0:
            MainMenu()
        else:
            print("Invalid Option")

def Customer_menu():
    while True:
        print('''
              --------------------------
              Customer menu:
              --------------------------
              1. Customer Registration
              2. Customer Search
              3. Customer Update
              0. Previous menu
              ''')
        C_option = int(input("Enter your option: "))
        if C_option == 1:
            cust_name = input("Enter Customer's name: ")
            cust_addr = input("Enter Customer's address: ")
            cust_city = input("Enter Customer's city: ")
            cust_state = input("Enter Customer's state: ")
            cust_email = input("Enter Customer's email: ")
            cust_phone = input("Enter Customer's phone number: ")
            print(f"Customer's name: {cust_name}\nCustomer's address:
 {cust_addr}\nCustomer's city: {cust_city}\nCustomer's state:
 {cust_state}\nCustomer's email: {cust_email}\nCustomer's phone number:
 {cust_phone}")
            c = conn.cursor()
```

```
            c.execute('''INSERT INTO Customer(cust_name, cust_addr, cust_city,
 cust_state, cust_email, cust_phone) VALUES (?, ?, ?, ?, ?, ?)''',(cust_name,
 cust_addr, cust_city, cust_state, cust_email, cust_phone))
            conn.commit()
        elif C_option == 2:
            CName = input("Enter Customer's name: ")
            CName = CName.upper()
            print('\n')
            c = conn.cursor()
            c.execute("SELECT customer_id, cust_name, cust_addr, cust_city,
 cust_state, cust_email, cust_phone FROM Customer WHERE upper(cust_name)=
 ?",(CName,))
            results =tabulate(c.fetchall(),headers=["cust_id", "cust_name",
 "cust_addr", "cust_city", "cust_state", "cust_email", "cust_phone"])
            if results:
                print(results)
                print('\n')
            conn.commit()
        elif C_option == 3:
            cust_identity = input("Enter Customer's id: ")
            cust_name = input("Enter updated customer's name: ")
            cust_addr = input("Enter updated customer's address: ")
            cust_city = input("Enter updated customer's city: ")
            cust_state = input("Enter updated customer's state: ")
            cust_email = input("Enter updated customer's email: ")
            cust_phone = input("Enter updated customer's phone number: ")
            c = conn.cursor()
            c.execute('''UPDATE Customer SET cust_name=?, cust_addr=?,
 cust_city=?, cust_state=?, cust_email=?, cust_phone=? WHERE customer_id
 =?''', (cust_name, cust_addr, cust_city, cust_state, cust_email, cust_phone,
 cust_identity))
            conn.commit()
        elif C_option == 0:
            MainMenu()
        else:
            print("Invalid Option")

def Customer_Order():
    while True:
        print('''
              --------------------------
```

```
            Customer order menu:
            --------------------------
            1. Order creation
            2. Order pending list
            3. Order search
            4. Order payment status update
            5. Order shipment list
            6. Shipment status search / update
            7. Order Invoice
            0. Previous menu
            ''')
        C_order_option = int(input("Enter your option: "))
        if C_order_option == 1:
            inp = 'y'
            inp = inp.lower()
            t_customer_id = int(input("Enter customer id: "))
            c = conn.cursor()
            c.execute("SELECT CAST(UNIXEPOCH() AS INTEGER)")
            t_order_id = c.fetchone()[0]
            c.execute('''CREATE TEMP TABLE Temp_cust_order(
                      order_id INTEGER,
                      product_id INTEGER,
                      quantity INTEGER,
                      sell_price REAL,
                      tax_rate REAL
            )''')
            while inp == 'y':
                t_product_id = int(input("Enter product id: "))
                t_quantity = int(input("Enter quantity of product: "))
                c.execute("SELECT sell_price FROM Product_Inventory WHERE
product_id= ?",(t_product_id,))
                t_sell_price = c.fetchone()[0]
                c.execute("SELECT tax_rate FROM Product_Inventory WHERE
product_id= ?", (t_product_id,))
                t_tax_rate = c.fetchone()[0]
                c.execute('''INSERT INTO Temp_cust_order(order_id, product_id,
quantity, sell_price, tax_rate) VALUES (?, ?, ?, ?, ?)''', (t_order_id,
t_product_id, t_quantity, t_sell_price, t_tax_rate))
                while True:
                    inp = input("Do you want to order more (y/n): ")
                    inp = inp.lower()
```

```
                    if inp in ['y', 'n']:
                        break
                    else:
                        print("Invalid input. Please enter y or n")
            c.execute('''INSERT INTO Customer_Order_Details(order_id,
product_id, quantity, sell_price, tax_rate) SELECT * FROM Temp_cust_order''')
            c.execute("SELECT SUM(quantity*sell_price*(1+tax_rate)) FROM
Temp_cust_order WHERE order_id= ?",(t_order_id,))
            t_total_amount = c.fetchone()[0]
            c.execute('''INSERT INTO Customer_order(customer_id, order_id,
total_amount) VALUES (?, ?, ?)''', (t_customer_id, t_order_id,
t_total_amount))
            c.execute("DROP TABLE Temp_cust_order")
            t_payment_status = "validating"
            t_payment_mode = "Online"
            c.execute("INSERT INTO Customer_Payment(customer_id, order_id,
payment_status, payment_mode) VALUES (?, ?, ?, ?)", (t_customer_id,
t_order_id, t_payment_status, t_payment_mode))
            conn.commit()
        elif C_order_option == 2:
            print('\n')
            c = conn.cursor()
            c.execute('''SELECT CP.customer_id, CP.order_id, CO.order_date,
CP.payment_status, CP.payment_mode FROM Customer_Payment CP,Customer_Order CO
WHERE CP.payment_status='validating' AND CP.order_id=CO.order_id LIMIT 5''')
            results =tabulate(c.fetchall(),headers=["customer_id", "order_id",
"order_date", "payment_status", "payment_mode"])
            if results:
                print(results)
                print('\n')
            conn.commit()
        elif C_order_option == 3:
            t_order_id = int(input("Enter order id: "))
            print('\n')
            c = conn.cursor()
            c.execute("SELECT customer_id, order_id, payment_id,
payment_status, payment_mode FROM Customer_payment WHERE order_id=
?",(t_order_id,))
            results =tabulate(c.fetchall(),headers=["customer_id", "order_id",
"payment_id", "payment_status", "payment_mode"])
            if results:
```

```
                print(results)
                print('\n')
            conn.commit()
        elif C_order_option == 4:
            t_order_id = int(input("Enter your order id: "))
            t_status = str(input("Is the payment done?(y/n): "))
            t_status = t_status.lower()
            if t_status == 'y':
                t_payment_status = "Success"
                t_shipment_status = "Order is being prepared..."
                t_tracking_reference = "Will be updated soon"
                t_provider_id = int(input("Enter the logistics provider id:
"))
                c = conn.cursor()
                c.execute('''SELECT warehouse_id FROM Product_Inventory PI,
Customer_Order_Details COD WHERE COD.product_id=PI.product_id AND order_id=
?''',(t_order_id,))
                t_warehouse_id = c.fetchone()[0]
                c.execute('''UPDATE Customer_Payment SET payment_status=?
WHERE order_id=?''',(t_payment_status, t_order_id))
                c.execute('''INSERT INTO Customer_Shipment(order_id,
shipment_status, warehouse_id, provider_id, tracking_reference) VALUES(?, ?,
?, ?, ?)''', (t_order_id, t_shipment_status, t_warehouse_id, t_provider_id,
t_tracking_reference))
                c.execute("SELECT product_id, quantity FROM
Customer_Order_details WHERE order_id= ?",(t_order_id,))
                records = c.fetchall()
                for row in records:
                    c.execute('''UPDATE Product_Inventory SET quantity=
quantity-? WHERE product_id=?''',(row[1], row[0]))
                conn.commit()
            else:
                t_payment_status = "Order Cancelled"
                c = conn.cursor()
                c.execute('''UPDATE Customer_Payment SET payment_status=?
WHERE order_id=?''',(t_payment_status, t_order_id))
                conn.commit()
        elif C_order_option == 5:
            print('\n')
            c = conn.cursor()
            t_shipment_status = "Order is shipped"
```

```
            c.execute("SELECT order_id, shipment_status, warehouse_id,
tracking_reference FROM Customer_Shipment WHERE shipment_status!= ? LIMIT
5",(t_shipment_status,))
            results =tabulate(c.fetchall(),headers=["order_id",
"shipment_status", "warehouse_id", "tracking_reference"])
            if results:
                print(results)
                print('\n')
            conn.commit()
        elif C_order_option == 6:
            t_order_id = int(input("Enter your order id: "))
            print('\n')
            c = conn.cursor()
            c.execute('''SELECT shipment_status FROM Customer_Shipment WHERE
order_id =?''',(t_order_id,))
            t_shipment_status = c.fetchone()[0]
            c.execute('''SELECT order_id, shipment_date, shipment_status,
provider_name, CS.provider_id, tracking_reference FROM Customer_Shipment CS,
Logistics_Provider LP WHERE CS.provider_id = LP.provider_id AND order_id
=?''',(t_order_id,))
            results =tabulate(c.fetchall(),headers=["order_id",
"shipment_date", "shipment_status", "provider_name", "provider_id",
"tracking_reference"])
            if results:
                print(results)
                print('\n')
            if t_shipment_status != "Order is shipped":
                t_status = str(input("Is the order ready for shipment?(y/n):
"))
                t_status = t_status.lower()
                if t_status == 'y':
                    t_shipment_status = "Order is shipped"
                    c = conn.cursor()
                    c.execute('''SELECT ABS(RANDOM()) AS tf ''')
                    t_tracking_reference = c.fetchone()[0]
                    c.execute('''SELECT CURRENT_TIMESTAMP''')
                    t_shipment_date = c.fetchone()[0]
                    t_provider_id = int(input("Enter the logistics provider
id: "))
                    c.execute('''UPDATE Customer_Shipment SET
shipment_status=?, shipment_date=?, provider_id=?, tracking_reference=? WHERE
```

```
order_id=?''',(t_shipment_status, t_shipment_date, t_provider_id,
t_tracking_reference, t_order_id))
            conn.commit()
        elif C_order_option == 7:
            t_order_id = int(input("Enter the invoice order id: "))
            print('\n')
            print("-------------------------------------------------------------
-------------------------------------")
            print(f"                                    Invoice for Order id:
{t_order_id}                              ")
            print("-------------------------------------------------------------
-------------------------------------")
            c = conn.cursor()
            c.execute('''SELECT cust_name, cust_addr, cust_city, cust_state,
cust_phone, order_id FROM Customer C, Customer_Order CO WHERE CO.customer_id=
C.customer_id AND order_id=?''',(t_order_id,))
            records = c.fetchall()
            for row in records:
                print("Name: ",row[0])
                print("Address: ", row[1])
                print("City: ",row[2])
                print("State: ", row[3])
                print("Phone no: ", row[4])
                print("Order id: ", row[5])
            print("-------------------------------------------------------------
-------------------------------------")
            c.execute('''SELECT product_name, COD.quantity, COD.sell_price,
COD.tax_rate, (COD.quantity*COD.sell_price*(1+COD.tax_rate)) subtotal FROM
Customer_Order_details COD, Product_Inventory PI WHERE COD.product_id=
PI.product_id AND order_id=?''',(t_order_id,))
            results =tabulate(c.fetchall(),headers=["product_name",
"quantity", "sell_price", "tax_rate", "sub_total"])
            if results:
                print(results)
            print('\n')
            c.execute('''SELECT order_date, delivery_date, total_amount FROM
Customer_Order WHERE order_id=?''',(t_order_id,))
            results =tabulate(c.fetchall(),headers=["order_date",
"delivery_date", "total_amount"])
            if results:
                print(results)
```

```
            conn.commit()
            print("-------------------------------------------------------------
-------------------------------------")
        elif C_order_option == 0:
            MainMenu()
        else:
            print("Invalid Option")

def Supplier_Order():
    while True:
        print('''
              --------------------------
              Supplier order menu:
              --------------------------
              1. Shortage of goods
              2. Order creation
              3. Supplier pending order
              4. Supplier order cancellation
              5. Supplier order receiving
              6. Order payment update
              0. Previous menu
              ''')
        Supp_order_option = int(input("Enter your option: "))
        if Supp_order_option == 1:
            t_quantity = 75
            c = conn.cursor()
            c.execute("SELECT product_id, product_name, quantity, supplier_id
FROM Product_Inventory WHERE quantity < ?", (t_quantity,))
            shortage_products = c.fetchall()
            if shortage_products:
                print("Shortage of Goods Report:")
                print('\n')
                headers = ["Prod_ID", "Prod_Name", "Quantity", "Supp_id"]
                print(tabulate(shortage_products, headers=headers))
            else:
                print("No shortage of goods")
            conn.commit()
        elif Supp_order_option == 2:
            t_product_id = int(input("Enter the product id: "))
            c = conn.cursor()
```

```
            c.execute('''SELECT supplier_id FROM Product_Inventory WHERE
product_id=?''',(t_product_id,))
            t_supplier_id = c.fetchone()[0]
            t_new_quantity = int(input("Enter the quantity of product: "))
            t_shipment_status = "Order submitted"
            c.execute('''INSERT INTO Supplier_Order(supplier_id, product_id,
quantity, shipment_status) VALUES (?, ?, ?, ?)''', (t_supplier_id,
t_product_id, t_new_quantity, t_shipment_status))
            conn.commit()
        elif Supp_order_option == 3:
            print("Supplier pending list")
            print('\n')
            t_shipment_status = "Order submitted"
            c = conn.cursor()
            c.execute('''SELECT supplier_id, sup_order_id, product_id,
quantity, shipment_status FROM Supplier_Order WHERE
shipment_status=?''',(t_shipment_status,))
            results =tabulate(c.fetchall(),headers=["supplier_id",
"supplier_order_id", "product_id", "quantity", "shipment_status"])
            if results:
                print(results)
                print('\n')
            conn.commit()
        elif Supp_order_option == 4:
            t_sup_order_id = int(input("Enter the supplier order id: "))
            print('\n')
            t_status = str(input("Do you want to cancel the order?(y/n): "))
            t_shipment_status = "Order cancelled"
            t_status = t_status.lower()
            if t_status == 'y':
                c = conn.cursor()
                c.execute('''UPDATE Supplier_Order SET shipment_status=? WHERE
sup_order_id=?''',(t_shipment_status, t_sup_order_id))
                conn.commit()
        elif Supp_order_option == 5:
            t_sup_order_id = int(input("Enter the supplier order id: "))
            t_shipment_status = "Order received"
            t_new_quantity = int(input("Enter the quantity of product: "))
            c.execute('''SELECT UNIXEPOCH()''')
            t_payment_reference = c.fetchone()[0]
            t_payment_status = "Payment Initiated"
```

```
            t_payment_mode = "Cheque payment"
            c = conn.cursor()
            c.execute('''SELECT product_id FROM Supplier_Order WHERE
sup_order_id=?''',(t_sup_order_id,))
            t_product_id = c.fetchone()[0]
            c.execute('''SELECT (buy_price*?) FROM Product_Inventory WHERE
product_id=?''',(t_new_quantity, t_product_id,))
            t_payment_amount = c.fetchone()[0]
            c.execute('''UPDATE Supplier_Order SET shipment_status=?,
quantity=? WHERE sup_order_id=?''',(t_shipment_status, t_new_quantity,
t_sup_order_id))
            c.execute('''INSERT INTO Supplier_Payment(sup_order_id,
payment_reference, payment_status, payment_mode, payment_amount) VALUES(?, ?,
?, ?, ?)''',(t_sup_order_id, t_payment_reference, t_payment_status,
t_payment_mode, t_payment_amount))
            c.execute('''UPDATE Product_Inventory SET quantity= quantity+?
WHERE product_id=?''',(t_new_quantity, t_product_id))
            conn.commit()
        elif Supp_order_option == 6:
            t_sup_order_id = int(input("Enter the supplier order id: "))
            t_payment_status = "Payment Initiated"
            c = conn.cursor()
            c.execute('''SELECT sup_order_id, payment_amount,
payment_reference, payment_duedate, payment_mode FROM Supplier_Payment WHERE
payment_status=? LIMIT 5''',(t_payment_status,))
            print('\n')
            results =tabulate(c.fetchall(),headers=["supplier_order_id",
"payment_amount", "payment_reference", "payment_duedate", "payment_mode"])
            if results:
                print(results)
                print('\n')
            t_payment_status = "Payment Completed"
            c.execute('''SELECT CURRENT_DATE''')
            t_payment_duedate= c.fetchone()[0]
            c.execute('''UPDATE Supplier_Payment SET payment_status=?,
payment_duedate=? WHERE sup_order_id=?''', (t_payment_status,
t_payment_duedate, t_sup_order_id))
            conn.commit()
        elif Supp_order_option == 0:
            MainMenu()
        else:
```

```
            print("Invalid Option")

def MainMenu():
    while True:
        print('''
          ----------------------
          Main Menu
          ----------------------
          1. Warehouse
          2. Supplier
          3. Logistics Provider
          4. Product Inventory
          5. Customer
          6. Customer's order
          7. Supplier's order
          0. Exit
          ''')
        option = int(input("Enter your option: "))
        if option == 1:
            Warehouse_menu()
        elif option == 2:
            Supplier_menu()
        elif option == 3:
            Logistics_menu()
        elif option == 4:
            Product_menu()
        elif option == 5:
            Customer_menu()
        elif option == 6:
            Customer_Order()
        elif option == 7:
            Supplier_Order()
        elif option == 0:
            conn.close()
            os._exit(0)
        else:
            print("Invalid option")

MainMenu()
#541 lines, using Visual Studio Code
```

Warehouse Sqlite3 Database: (warehouse.sql)

```
CREATE TABLE IF NOT EXISTS Customer(
        customer_id INTEGER PRIMARY KEY AUTOINCREMENT,
        cust_name TEXT NOT NULL,
        cust_addr TEXT NOT NULL,
        cust_city TEXT NOT NULL,
        cust_state TEXT NOT NULL,
        cust_email TEXT NOT NULL,
        cust_phone TEXT NOT NULL
);
CREATE TABLE IF NOT EXISTS Customer_Order_Details(
        sub_order_id INTEGER PRIMARY KEY AUTOINCREMENT,
        order_id INTEGER NOT NULL,
        product_id INTEGER NOT NULL,
        sell_price REAL NOT NULL,
        quantity INTEGER NOT NULL,
        tax_rate REAL NOT NULL
);
CREATE TABLE IF NOT EXISTS Customer_Order(
        customer_id INTEGER NOT NULL,
        order_id INTEGER PRIMARY KEY,
        order_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
        delivery_date DATE DEFAULT (date(CURRENT_DATE,'+7 days')),
        total_amount REAL NOT NULL,
        FOREIGN KEY(customer_id) REFERENCES Customer(customer_id),
        FOREIGN KEY(order_id) REFERENCES Customer_Order_Details(order_id)
);
CREATE TABLE IF NOT EXISTS Customer_Payment(
        customer_id INTEGER NOT NULL,
        order_id INTEGER NOT NULL,
        payment_id INTEGER PRIMARY KEY AUTOINCREMENT,
        payment_status TEXT,
        payment_mode TEXT,
        FOREIGN KEY(customer_id) REFERENCES Customer(customer_id),
        FOREIGN KEY(order_id) REFERENCES Customer_Order(order_id)
);
CREATE TABLE IF NOT EXISTS Warehouse(
        warehouse_id INTEGER PRIMARY KEY AUTOINCREMENT,
        warehouse_address TEXT NOT NULL,
```

```sql
        warehouse_city TEXT NOT NULL,
        warehouse_state TEXT NOT NULL,
        warehouse_email TEXT NOT NULL,
        warehouse_phone TEXT NOT NULL
);
CREATE TABLE IF NOT EXISTS Logistics_Provider(
        provider_id INTEGER PRIMARY KEY AUTOINCREMENT,
        provider_name TEXT NOT NULL,
        provider_address TEXT NOT NULL,
        provider_phone TEXT NOT NULL,
        provider_email TEXT NOT NULL,
        tracking_url TEXT NOT NULL
);
CREATE TABLE IF NOT EXISTS Customer_Shipment(
        shipment_id INTEGER PRIMARY KEY AUTOINCREMENT,
        order_id INTEGER NOT NULL,
        shipment_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
        shipment_status TEXT NOT NULL,
        warehouse_id INTEGER NOT NULL,
        provider_id INTEGER NOT NULL,
        tracking_reference TEXT NOT NULL,
        FOREIGN KEY(order_id) REFERENCES Customer_Order(order_id),
        FOREIGN KEY(provider_id) REFERENCES Logistics_Provider(provider_id),
        FOREIGN KEY(warehouse_id) REFERENCES Warehouse(warehouse_id)
);
CREATE TABLE IF NOT EXISTS Supplier(
        supplier_id INTEGER PRIMARY KEY AUTOINCREMENT,
        supplier_name TEXT NOT NULL,
        supplier_address TEXT NOT NULL,
        supplier_city TEXT NOT NULL,
        contact_person TEXT NOT NULL,
        supplier_email TEXT NOT NULL,
        supplier_phone TEXT NOT NULL
);
CREATE TABLE IF NOT EXISTS Product_Inventory(
        warehouse_id INTEGER NOT NULL,
        product_id INTEGER PRIMARY KEY AUTOINCREMENT,
        product_name TEXT NOT NULL,
        product_descrip TEXT NOT NULL,
        buy_price REAL NOT NULL,
        sell_price REAL NOT NULL,
```

```
        tax_rate REAL NOT NULL,
        quantity INTEGER NOT NULL,
        category TEXT NOT NULL,
        supplier_id INTEGER NOT NULL,
        FOREIGN KEY(warehouse_id) REFERENCES Warehouse(warehouse_id),
        FOREIGN KEY(supplier_id) REFERENCES Supplier(supplier_id)
);
CREATE TABLE IF NOT EXISTS Supplier_Order(
        supplier_id INTEGER NOT NULL,
        sup_order_id INTEGER PRIMARY KEY AUTOINCREMENT,
        product_id INTEGER NOT NULL,
        quantity INTEGER NOT NULL,
        order_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
        delivery_date DATE DEFAULT (date(CURRENT_DATE,'+7 days')),
        shipment_status TEXT NOT NULL,
        FOREIGN KEY(supplier_id) REFERENCES Supplier(supplier_id),
        FOREIGN KEY(product_id) REFERENCES Product_Inventory(product_id)
);
CREATE TABLE IF NOT EXISTS Supplier_Payment(
        sup_order_id INTEGER NOT NULL,
        payment_id INTEGER PRIMARY KEY AUTOINCREMENT,
        payment_amount REAL NOT NULL,
        payment_reference TEXT NOT NULL,
        payment_duedate DATE DEFAULT (date(CURRENT_DATE,'+60 days')),
        payment_status TEXT NOT NULL,
        payment_mode TEXT NOT NULL,
        FOREIGN KEY(sup_order_id) REFERENCES Supplier_Order(sup_order_id)
);
```

OUTPUT

Main Menu & Warehouse menu():

- To access the application menu issue the command as described below:
- To access the submenu select the respective number options and press enter.

```
Windows PowerShell

PS C:\Users\haris\Desktop\WMS_project\warehouse> python.exe warehouse.py

          ----------------------
          Main Menu
          ----------------------
          1. Warehouse
          2. Supplier
          3. Logistics Provider
          4. Product Inventory
          5. Customer
          6. Customer's order
          7. Supplier's order
          0. Exit

Enter your option: 1

              --------------------------
              Warehouse menu:
              --------------------------
              1. Warehouse Registration
              2. Warehouse Search
              3. Warehouse Update
              0. Previous menu

Enter your option:
```

Warehouse sub-menu:

- New warehouses can be registered using option 1.

```
Enter your option: 1

              --------------------------
              Warehouse menu:
              --------------------------
              1. Warehouse Registration
              2. Warehouse Search
              3. Warehouse Update
              0. Previous menu

Enter your option: 1
Enter Warehouse address: 16 Westshore ave 7
Enter location of warehouse city: Mangalore
Enter the state of warehouse: Karnataka
Enter warehouse's e-mail address: mang@warehouse.com
Enter warehouse's phone number: 9875673498
Warehouse Address: 16 Westshore ave 7
Warehouse City: Mangalore
Warehouse State: Karnataka
Warehouse Email: mang@warehouse.com
Warehouse Phone: 9875673498

              --------------------------
              Warehouse menu:
              --------------------------
              1. Warehouse Registration
              2. Warehouse Search
              3. Warehouse Update
              0. Previous menu

Enter your option:
```

- Existing warehouse details can be searched using option 2 as shown below.

```
              --------------------------
              Warehouse menu:
              --------------------------
              1. Warehouse Registration
              2. Warehouse Search
              3. Warehouse Update
              0. Previous menu

Enter your option: 2
Enter the warehouse city: Mangalore


  warehouse_id  warehouse_address    warehouse_city    warehouse_state    warehouse_email       warehouse_phone
--------------  -------------------  ----------------  -----------------  ------------------  -----------------
            12  16 Westshore ave 7   Mangalore         Karnataka          mang@warehouse.com         9875673498



              --------------------------
              Warehouse menu:
              --------------------------
              1. Warehouse Registration
              2. Warehouse Search
              3. Warehouse Update
              0. Previous menu

Enter your option:
```

❖ Existing warehouse details can be updated using option 3 as shown below.

```
  warehouse_id  warehouse_address    warehouse_city    warehouse_state    warehouse_email       warehouse_phone
--------------  -------------------  ----------------  -----------------  ------------------  -----------------
            12  16 Westshore ave 7   Mangalore         Karnataka          mang@warehouse.com         9875673498



              --------------------------
              Warehouse menu:
              --------------------------
              1. Warehouse Registration
              2. Warehouse Search
              3. Warehouse Update
              0. Previous menu

Enter your option: 3
Enter Warehouse id: 12
Enter Warehouse address: 16 Westshore ave 7
Enter location of warehouse city: Mangalore
Enter the state of warehouse: Karnataka
Enter warehouse's e-mail address: mang@warehouse.com
Enter warehouse's phone number: 8877665544

              --------------------------
              Warehouse menu:
              --------------------------
              1. Warehouse Registration
              2. Warehouse Search
              3. Warehouse Update
```

❖ The updated warehouse details can be verified by using option 2 as shown below.

```
              -------------------------
              Warehouse menu:
              -------------------------
              1. Warehouse Registration
              2. Warehouse Search
              3. Warehouse Update
              0. Previous menu

Enter your option: 2
Enter the warehouse city: Mangalore


  warehouse_id  warehouse_address    warehouse_city    warehouse_state    warehouse_email       warehouse_phone
--------------  -------------------  ----------------  -----------------  ------------------  -----------------
            12  16 Westshore ave 7   Mangalore         Karnataka          mang@warehouse.com         8877665544



              -------------------------
              Warehouse menu:
              -------------------------
              1. Warehouse Registration
              2. Warehouse Search
              3. Warehouse Update
              0. Previous menu

Enter your option:
```

Supplier sub-menu:

- By selecting option 0 we can exit the warehouse menu and return back to the Main-Menu.
- By selecting option 2 we can move on to the Supplier Menu.

```
          ---------------------
          Main Menu
          ---------------------
          1. Warehouse
          2. Supplier
          3. Logistics Provider
          4. Product Inventory
          5. Customer
          6. Customer's order
          7. Supplier's order
          0. Exit

Enter your option: 2

              -------------------------
              Supplier menu:
              -------------------------
              1. Supplier Registration
              2. Supplier Search
              3. Supplier Update
              0. Previous menu

Enter your option: |
```

- A new supplier can be registered as shown below.

```
Enter your option: 2

              -------------------------
              Supplier menu:
              -------------------------
              1. Supplier Registration
              2. Supplier Search
              3. Supplier Update
              0. Previous menu

Enter your option: 1
Enter supplier's name: Philip Electricals
Enter contact person name for supplier: Ramanujam
Enter supplier's e-mail address: ramjam@philip.com
Enter supplier's phone number: 9955445566
Enter supplier's address: 13 Gift city
Enter supplier's city: Ahmedabad metro
Supplier Name: Philip Electricals
Contact Person: Ramanujam
Supplier Email: ramjam@philip.com
Supplier Phone: 9955445566
Supplier Address: 13 Gift city
Supplier City: Ahmedabad metro

              -------------------------
              Supplier menu:
              -------------------------
              1. Supplier Registration
              2. Supplier Search
              3. Supplier Update
              0. Previous menu

Enter your option:
```

- The registered new supplier can be searched by ‘name’ using option 2 as shown below.

```
Enter Supplier id: 12
Enter Supplier address: 13 Gift city
Enter Supplier city: Ahmedabad
Enter Supplier's e-mail address: ram@philip.com
Enter Supplier's phone number: 9667788993

              --------------------------
              Supplier menu:
              --------------------------
              1. Supplier Registration
              2. Supplier Search
              3. Supplier Update
              0. Previous menu

Enter your option: 2
Enter supplier's name: Philip Electricals


  supplier_id  supplier_name       supplier_address    supplier_city    contact_person    supplier_email      supplier_phone
-------------  ------------------  ------------------  ---------------  ----------------  ----------------  ----------------
           12  Philip Electricals  13 Gift city        Ahmedabad        Ramanujam         ram@philip.com          9667788993


              --------------------------
              Supplier menu:
              --------------------------
              1. Supplier Registration
              2. Supplier Search
              3. Supplier Update
              0. Previous menu

Enter your option:
```

❖ The supplier's details can be updated by using option 3 as shown below.

```
Enter supplier's name: Philip Electricals


  supplier_id  supplier_name       supplier_address    supplier_city    contact_person    supplier_email      supplier_phone
-------------  ------------------  ------------------  ---------------  ----------------  ----------------  ----------------
           12  Philip Electricals  13 Gift city        Ahmedabad metro  Ramanujam         ramjam@philip.com       9955445566


              --------------------------
              Supplier menu:
              --------------------------
              1. Supplier Registration
              2. Supplier Search
              3. Supplier Update
              0. Previous menu

Enter your option: 3
Enter Supplier id: 12
Enter Supplier address: 13 Gift city
Enter Supplier city: Ahmedabad
Enter Supplier's e-mail address: ram@philip.com
Enter Supplier's phone number: 9667788993

              --------------------------
              Supplier menu:
              --------------------------
              1. Supplier Registration
              2. Supplier Search
              3. Supplier Update
              0. Previous menu

Enter your option:
```

❖ The updated supplier's details can be searched by using option 2 as shown below.

```
Enter Supplier id: 12
Enter Supplier address: 13 Gift city
Enter Supplier city: Ahmedabad
Enter Supplier's e-mail address: ram@philip.com
Enter Supplier's phone number: 9667788993

              -------------------------
              Supplier menu:
              -------------------------
              1. Supplier Registration
              2. Supplier Search
              3. Supplier Update
              0. Previous menu

Enter your option: 2
Enter supplier's name: Philip Electricals


  supplier_id  supplier_name       supplier_address    supplier_city    contact_person    supplier_email      supplier_phone
-------------  -----------------  -----------------  --------------  ----------------  ----------------  ----------------
           12  Philip Electricals  13 Gift city       Ahmedabad       Ramanujam         ram@philip.com       9667788993


              -------------------------
              Supplier menu:
              -------------------------
              1. Supplier Registration
              2. Supplier Search
              3. Supplier Update
              0. Previous menu

Enter your option:
```

❖ To go back to the main menu select option 0.

Logistics Provider sub-menu:

❖ The logistics provider option is selected by using option 3.

```
          -----------------------
          Main Menu
          -----------------------
          1. Warehouse
          2. Supplier
          3. Logistics Provider
          4. Product Inventory
          5. Customer
          6. Customer's order
          7. Supplier's order
          0. Exit

Enter your option: 3

              -------------------------
              Logistics menu:
              -------------------------
              1. Logistics Registration
              2. Logistics Search
              3. Logistics Update
              0. Previous menu

Enter your option:
```

- To register a new logistics provider, option 1 is selected as shown below.

```
Enter your option: 3

              --------------------------
              Logistics menu:
              --------------------------
              1. Logistics Registration
              2. Logistics Search
              3. Logistics Update
              0. Previous menu

Enter your option: 1
Enter Logistics provider's name: Maruti couriers
Enter Logistics provider's address: 12 Commercial st
Enter Logistics provider's phone number: 08078654876
Enter Logistics provider's e-mail address: blr@maruti.com
Enter Logistics provider's tracking url: maruti.com
Logistics provider's name: Maruti couriers
Logistics provider's address: 12 Commercial st
Logistics provider's phone: 08078654876
Logistics provider's email: blr@maruti.com
Logistics provider's tracking url: maruti.com

              --------------------------
              Logistics menu:
              --------------------------
              1. Logistics Registration
              2. Logistics Search
              3. Logistics Update
              0. Previous menu

Enter your option:
```

- In a similar way (as described above for the warehouse and supplier menu) the logistics provider details can be searched and updated by using option 2 & 3 respectively.
- To move on to the next menu (Product_Inventory) option 0 is selected.
- Product registration is done by choosing option 1 as described below.

Product registration sub-menu:

```
Enter your option: 4

               --------------------------
               Product menu:
               --------------------------
               1. Product Registration
               2. Product Search
               3. Product Update
               0. Previous menu

Enter your option: 1
Enter product's name: JBL flip 2
Enter product's description: IPX7 20W
Enter product's buying price: 5000
Enter product's tax rate: 0.18
Enter quantity of product: 50
Enter category of product: Speaker
Enter supplier id: 7
Enter warehouse id: 3
Product's name: JBL flip 2
Product's description: IPX7 20W
Product's buying price: 5000.0
Product's selling price: 6250.0
Product's tax rate: 0.18
Quantity of product: 50
Category of product: Speaker
Supplier id: 7
Warehouse id: 3

               --------------------------
               Product menu:
               --------------------------
               1. Product Registration
               2. Product Search
               3. Product Update
               0. Previous menu

Enter your option:
```

- Product's details can be searched (using category) and updated using option 2 and 3 respectively.
- To move on to the next menu (Customer) press 0 and select option 5 from the Main menu.

Customer registration sub-menu:

```
Enter your option: 5

              -------------------------
              Customer menu:
              -------------------------
              1. Customer Registration
              2. Customer Search
              3. Customer Update
              0. Previous menu

Enter your option: 1
Enter Customer's name: Tharun
Enter Customer's address: 144, parkinson st
Enter Customer's city: Hyderabad
Enter Customer's state: Telangana
Enter Customer's email: tharun@abc.com
Enter Customer's phone number: 0801295423
Customer's name: Tharun
Customer's address: 144, parkinson st
Customer's city: Hyderabad
Customer's state: Telangana
Customer's email: tharun@abc.com
Customer's phone number: 0801295423

              -------------------------
              Customer menu:
              -------------------------
              1. Customer Registration
              2. Customer Search
              3. Customer Update
              0. Previous menu

Enter your option: |
```

- In a similar way (as described above for the warehouse and supplier menu) the customer details can be searched and updated by using option 2 & 3 respectively.
- To move on to the next menu (Customer Order) option 0 is selected.
- Customers can place their order by choosing option 1 as described below.

Customer Order Placement sub-menu:

```
Enter your option: 6

               --------------------------
               Customer order menu:
               --------------------------
               1. Order creation
               2. Order pending list
               3. Order search
               4. Order payment status update
               5. Order shipment list
               6. Shipment status search / update
               7. Order Invoice
               0. Previous menu

Enter your option: 1
Enter customer id: 8
Enter product id: 7
Enter quantity of product: 2
Do you want to order more (y/n): y
Enter product id: 8
Enter quantity of product: 2
Do you want to order more (y/n): n

               --------------------------
               Customer order menu:
               --------------------------
               1. Order creation
               2. Order pending list
               3. Order search
               4. Order payment status update
               5. Order shipment list
               6. Shipment status search / update
               7. Order Invoice
               0. Previous menu

Enter your option: |
```

- To check customer order pending list option 2 is selected as shown below.

```
              --------------------------
              Customer order menu:
              --------------------------
              1. Order creation
              2. Order pending list
              3. Order search
              4. Order payment status update
              5. Order shipment list
              6. Shipment status search / update
              7. Order Invoice
              0. Previous menu

Enter your option: 2


  customer_id    order_id  order_date           payment_status    payment_mode
-------------  ----------  -------------------  ----------------  --------------
            4  1714550101  2024-05-01 07:55:11  validating        Online
            8  1714840754  2024-05-04 16:40:23  validating        Online



              --------------------------
              Customer order menu:
              --------------------------
              1. Order creation
              2. Order pending list
              3. Order search
              4. Order payment status update
              5. Order shipment list
              6. Shipment status search / update
              7. Order Invoice
              0. Previous menu

Enter your option: |
```

- To search for the customer's order select option 3 as described below.

```
              --------------------------
              Customer order menu:
              --------------------------
              1. Order creation
              2. Order pending list
              3. Order search
              4. Order payment status update
              5. Order shipment list
              6. Shipment status search / update
              7. Order Invoice
              0. Previous menu

Enter your option: 3
Enter order id: 1714840754


  customer_id    order_id    payment_id  payment_status    payment_mode
-------------  ----------  ------------  ----------------  --------------
            8  1714840754            13  validating        Online



              --------------------------
              Customer order menu:
              --------------------------
              1. Order creation
              2. Order pending list
              3. Order search
              4. Order payment status update
              5. Order shipment list
              6. Shipment status search / update
              7. Order Invoice
              0. Previous menu

Enter your option: |
```

- To proceed with payment for the customer's order select option 4 as described below.
- It also asks the customer to choose the logistics provider id so that delivery company can be selected by the customer.

```
              --------------------------
              Customer order menu:
              --------------------------
              1. Order creation
              2. Order pending list
              3. Order search
              4. Order payment status update
              5. Order shipment list
              6. Shipment status search / update
              7. Order Invoice
              0. Previous menu

Enter your option: 4
Enter your order id: 1714840754
Is the payment done?(y/n): y
Enter the logistics provider id: 3

              --------------------------
              Customer order menu:
              --------------------------
              1. Order creation
              2. Order pending list
              3. Order search
              4. Order payment status update
              5. Order shipment list
              6. Shipment status search / update
              7. Order Invoice
              0. Previous menu

Enter your option: |
```

❖ To check the order shipment list select option 5 as described below.

```
              --------------------------
              Customer order menu:
              --------------------------
              1. Order creation
              2. Order pending list
              3. Order search
              4. Order payment status update
              5. Order shipment list
              6. Shipment status search / update
              7. Order Invoice
              0. Previous menu

Enter your option: 5


  order_id  shipment_status              warehouse_id  tracking_reference
----------  -------------------------  --------------  --------------------
1714840754  Order is being prepared...               3  Will be updated soon



              --------------------------
              Customer order menu:
              --------------------------
              1. Order creation
              2. Order pending list
              3. Order search
              4. Order payment status update
              5. Order shipment list
              6. Shipment status search / update
              7. Order Invoice
              0. Previous menu

Enter your option: |
```

❖ To check the order shipment status or to update the shipment status select option 6 as described below.

```
              -------------------------
              Customer order menu:
              -------------------------
              1. Order creation
              2. Order pending list
              3. Order search
              4. Order payment status update
              5. Order shipment list
              6. Shipment status search / update
              7. Order Invoice
              0. Previous menu

Enter your option: 6
Enter your order id: 1714840754


  order_id  shipment_date        shipment_status            provider_name      provider_id  tracking_reference
----------  -------------------  -------------------------  ---------------  -------------  --------------------
1714840754  2024-05-04 16:42:35  Order is being prepared...  Speed Delivery               3  Will be updated soon


Is the order ready for shipment?(y/n): |
```

```
  order_id  shipment_date        shipment_status            provider_name      provider_id  tracking_reference
----------  -------------------  -------------------------  ---------------  -------------  --------------------
1714840754  2024-05-04 16:42:35  Order is being prepared...  Speed Delivery               3  Will be updated soon


Is the order ready for shipment?(y/n): y
Enter the logistics provider id: 4

              -------------------------
              Customer order menu:
              -------------------------
              1. Order creation
              2. Order pending list
              3. Order search
              4. Order payment status update
              5. Order shipment list
              6. Shipment status search / update
              7. Order Invoice
              0. Previous menu

Enter your option: |
```

Changing the provider id

❖ Now the order pending list must be cleared as we have completed the payment, to check that select option 2 as done above.

```
Enter your option: 4
Enter your order id: 1714550101
Is the payment done?(y/n): y
Enter the logistics provider id: 3

              -------------------------
              Customer order menu:
              -------------------------
              1. Order creation
              2. Order pending list
              3. Order search
              4. Order payment status update
              5. Order shipment list
              6. Shipment status search / update
              7. Order Invoice
              0. Previous menu

Enter your option: 2


customer_id    order_id    order_date    payment_status    payment_mode
-------------  ----------  ------------  ----------------  --------------



              -------------------------
              Customer order menu:
              -------------------------
              1. Order creation
              2. Order pending list
              3. Order search
              4. Order payment status update
              5. Order shipment list
              6. Shipment status search / update
              7. Order Invoice
              0. Previous menu

Enter your option: |
```

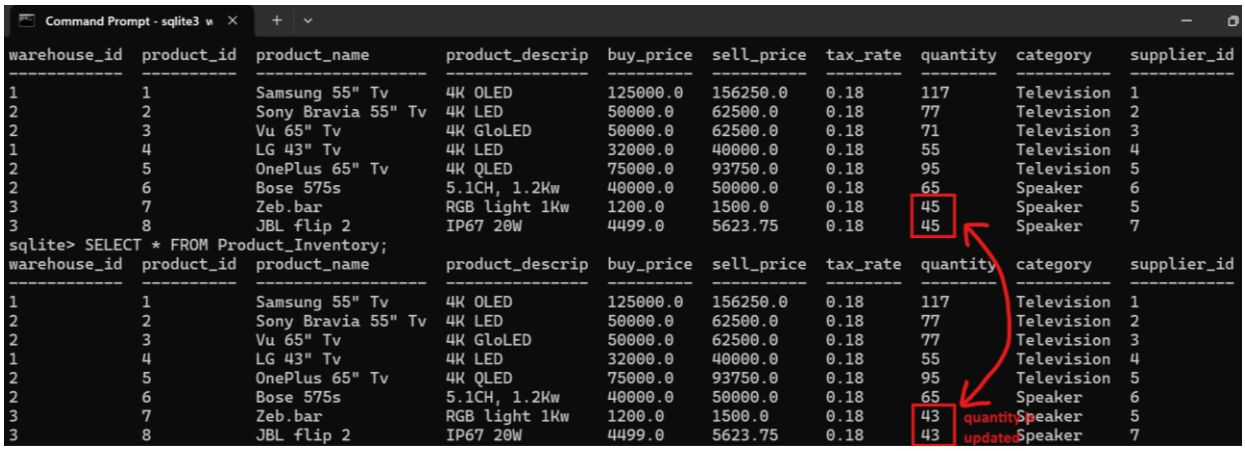
❖ Pending list has been cleared. Now to generate Order Invoice select option 7.

```
Enter your option: 7
Enter the invoice order id: 1714840754


-------------------------------------------------------------------------------------------
                                Invoice for Order id: 1714840754
-------------------------------------------------------------------------------------------
Name:  Tharunkumar
Address:  144, parkison st
City:  Hyderabad
State:  Telangana
Phone no:  0801234567
Order id:  1714840754
-------------------------------------------------------------------------------------------
product_name      quantity    sell_price    tax_rate    sub_total
--------------  ----------  ------------  ----------  -----------
Zeb.bar                  2       1500          0.18         3540
JBL flip 2               2       5623.75       0.18        13272


order_date           delivery_date      total_amount
-------------------  ---------------  --------------
2024-05-04 16:40:23  2024-05-11                16812
-------------------------------------------------------------------------------------------
```

❖ Now by checking the database we can infer that the count of products has decreased as the customer has ordered, which is described below.



Command Prompt - sqlite3 w

warehouse_id	product_id	product_name	product_descrip	buy_price	sell_price	tax_rate	quantity	category	supplier_id
1	1	Samsung 55" Tv	4K OLED	125000.0	156250.0	0.18	117	Television	1
2	2	Sony Bravia 55" Tv	4K LED	50000.0	62500.0	0.18	77	Television	2
2	3	Vu 65" Tv	4K GloLED	50000.0	62500.0	0.18	71	Television	3
1	4	LG 43" Tv	4K LED	32000.0	40000.0	0.18	55	Television	4
2	5	OnePlus 65" Tv	4K QLED	75000.0	93750.0	0.18	95	Television	5
2	6	Bose 575s	5.1CH, 1.2Kw	40000.0	50000.0	0.18	65	Speaker	6
3	7	Zeb.bar	RGB light 1Kw	1200.0	1500.0	0.18	45	Speaker	5
3	8	JBL flip 2	IP67 20W	4499.0	5623.75	0.18	45	Speaker	7

```
sqlite> SELECT * FROM Product_Inventory;
```

warehouse_id	product_id	product_name	product_descrip	buy_price	sell_price	tax_rate	quantity	category	supplier_id
1	1	Samsung 55" Tv	4K OLED	125000.0	156250.0	0.18	117	Television	1
2	2	Sony Bravia 55" Tv	4K LED	50000.0	62500.0	0.18	77	Television	2
2	3	Vu 65" Tv	4K GloLED	50000.0	62500.0	0.18	77	Television	3
1	4	LG 43" Tv	4K LED	32000.0	40000.0	0.18	55	Television	4
2	5	OnePlus 65" Tv	4K QLED	75000.0	93750.0	0.18	95	Television	5
2	6	Bose 575s	5.1CH, 1.2Kw	40000.0	50000.0	0.18	65	Speaker	6
3	7	Zeb.bar	RGB light 1Kw	1200.0	1500.0	0.18	43	Speaker	5
3	8	JBL flip 2	IP67 20W	4499.0	5623.75	0.18	43	Speaker	7

quantity updated

- To move on to supplier order select option 7 from main menu as well as exit from the customer order submenu which is described below.
- This process is initiated by Warehouse (not by customer).

Supplier Order sub-menu:

```
              -------------------------
              Customer order menu:
              -------------------------
              1. Order creation
              2. Order pending list
              3. Order search
              4. Order payment status update
              5. Order shipment list
              6. Shipment status search / update
              7. Order Invoice
              0. Previous menu

Enter your option: 0

          ----------------------
          Main Menu
          ----------------------
          1. Warehouse
          2. Supplier
          3. Logistics Provider
          4. Product Inventory
          5. Customer
          6. Customer's order
          7. Supplier's order
          0. Exit

Enter your option: 7

              -------------------------
              Supplier order menu:
              -------------------------
              1. Shortage of goods
              2. Order creation
              3. Supplier pending order
              4. Supplier order cancellation
              5. Supplier order receiving
              6. Order payment update
              0. Previous menu

Enter your option: |
```

- To check the shortage of goods select option 1 which is described below.

```
              --------------------------
              Supplier order menu:
              --------------------------
              1. Shortage of goods
              2. Order creation
              3. Supplier pending order
              4. Supplier order cancellation
              5. Supplier order receiving
              6. Order payment update
              0. Previous menu

Enter your option: 1
Shortage of Goods Report:


  Prod_ID  Prod_Name      Quantity    Supp_id
---------  -----------  ----------  ---------
        3  Vu 65" Tv            69          3
        4  LG 43" Tv            55          4
        6  Bose 575s            65          6
        7  Zeb.bar              43          5
        8  JBL flip 2           43          7

              --------------------------
              Supplier order menu:
              --------------------------
              1. Shortage of goods
              2. Order creation
              3. Supplier pending order
              4. Supplier order cancellation
              5. Supplier order receiving
              6. Order payment update
              0. Previous menu

Enter your option: |
```

- Order is placed for the mentioned shortage of goods in the warehouse.
- Similar to customer order placement process, supplier order is also done by selecting appropriate options.

CONCLUSION

Through the development of the Warehouse Management System project, we have acquired a wealth of knowledge and skills that significantly enriched our understanding of software development and database management. Here's a summary of what we learned:

1. Database Management: We gained hands-on experience in working with relational databases, particularly SQLite. This involved creating database schemas, defining tables, and implementing CRUD (Create, Read, Update, Delete) operations using SQL queries.
2. Python Programming: The project provided an opportunity to enhance our proficiency in Python programming. We have learned how to use Python for various tasks, including database interaction and input validation. We have improved our understanding of core concepts such as variables, loops, conditionals, functions and exception handling.
3. Data Validation: Ensuring the accuracy and integrity of user input data was essential for the system's reliability. It includes various techniques for validating user inputs, including type checking, range validation and format validation. We also learned how to handle input errors gracefully and provide helpful error messages to users.
4. Agile Development Methodologies: Throughout the project, we have learned the importance of breaking down tasks into smaller, manageable units and prioritizing features based on user needs.

Overall, the Warehouse Management System project was a transformative learning experience that equipped us with practical skills, theoretical knowledge and a deeper understanding of software development principles.

FUTURE ENHANCEMENTS

1. Integration with Barcode Scanning Technology:
    - Implementing barcode scanning functionality would streamline inventory management processes by allowing users to quickly identify and track products.
    - By scanning barcodes, warehouse staff can easily update inventory levels, record stock movements and perform stocktaking tasks with greater accuracy and efficiency.
2. Real-time Reporting and Analytics:
    - Enhance the system to generate real-time reports and analytics on key performance metrics such as sales trends, inventory turnover and supplier performance.
    - By providing actionable insights into warehouse operations, managers can make data-driven decisions to optimize inventory levels, streamline supply chain processes, and identify areas for improvement.
3. Mobile Application Support:
    - Develop a mobile application that complements the existing desktop-based system, allowing users to access critical warehouse management functionalities from their smartphones or tablets.
    - With mobile app support, warehouse staff can perform tasks such as inventory lookup, order processing, and receiving shipments on the go, enhancing operational flexibility and responsiveness.
4. Predictive Inventory Management:
    - Implement predictive analytics algorithms to forecast demand and optimize inventory replenishment strategies.
    - By analyzing historical sales data, seasonal trends, and market dynamics, the system can generate accurate demand forecasts, helping warehouse managers maintain optimal inventory levels and avoid stock-outs or overstock situations.

BIBLIOGRAPHY

1. Python Documentation:
    - Documentation for the Python programming language.
    - https://docs.python.org/3/library/index.html
2. SQLite Documentation:
    - Documentation for SQLite, the relational database management system used in this project.
    - https://www.sqlite.org/doclist.html
3. Tabulate Documentation:
    - Documentation for the Tabulate library used for formatting query results in a tabular format.
    - https://pypi.org/project/tabulate/
4. Stack overflow:
    - With millions of active users and an extensive database of questions and answers, Stack Overflow facilitates peer-to-peer learning and problem-solving in diverse programming languages, frameworks and technologies.
    - https://stackoverflow.com/questions