Name : Sarthak Shandilya

Python Assignment 1

In this assignment the program can be started with app.py

If you run the program you'll get 10 options and you have to select choices accordingly from 1 to 10 as shown in the program.

1. Customer Registration
2. Update products info
3. Placing new order
4. Track order status
5. Inventory management
6. Sales reporting
7. Customer Account Update (phone, address, email)
8. Payment Processing
9. Product search with recommendations based on category.
10. Exit from application.

App.py

```python
from DBUtil import DBUtil
from TechShopImpl import TechShopServiceImpl


1 usage
class TechShop(TechShopServiceImpl):
    def __init__(self, dbUtil):
        super().__init__(dbUtil)

    1 usage
    def main(self):
        while True:
            print("Please select the choices from below : ")
            print("1. Customer Registration. ")
            print("2. Update Product Info.")
            print("3. Place New Order")
            print("4. Tracking Order Status")
            print("5. Inventory Management")
            print("6. Sales Reporting")
            print("7. Customer Account Updates")
            print("8. Payment Processing")
            print("9. Product Search and Recommendations")
            print("10. Exit")
            choice = int(input("Enter your choice here : "))
            match choice:
                case 1:
                    self.user_registration()
                    print("Thank you for registering with us.")
                    print("We're heading you to main menu.")
                case 2:
                    self.changes_in_products()
```

```python
                    print("Thank you for registering with us.")
                    print("We're heading you to main menu.")
                case 2:
                    self.changes_in_products()
                    print("Changes made successfully!")
                    print("We're heading you to main menu.")
                case 3:
                    self.placing_order()
                    print("Order placed successfully.")
                    print("We're heading you to main menu.")
                case 4:
                    self.get_order_status()
                    print("We're heading you to main menu.")
                case 5:
                    self.manage_inventory()
                    print("We're heading you to main menu.")
                case 6:
                    self.report_sales()
                    print("We're heading you to main menu.")
                case 7:
                    self.customer_updates()
                    print("Updates were successful.")
                    print("We're heading you to main menu.")
                case 8:
                    self.process_payments()
                    print("We're heading you to main menu.")
                case 9:
                    self.search_products()
                    print()
                case 10:
                    print("Thanks for visiting us. See you anytime soon.")
```

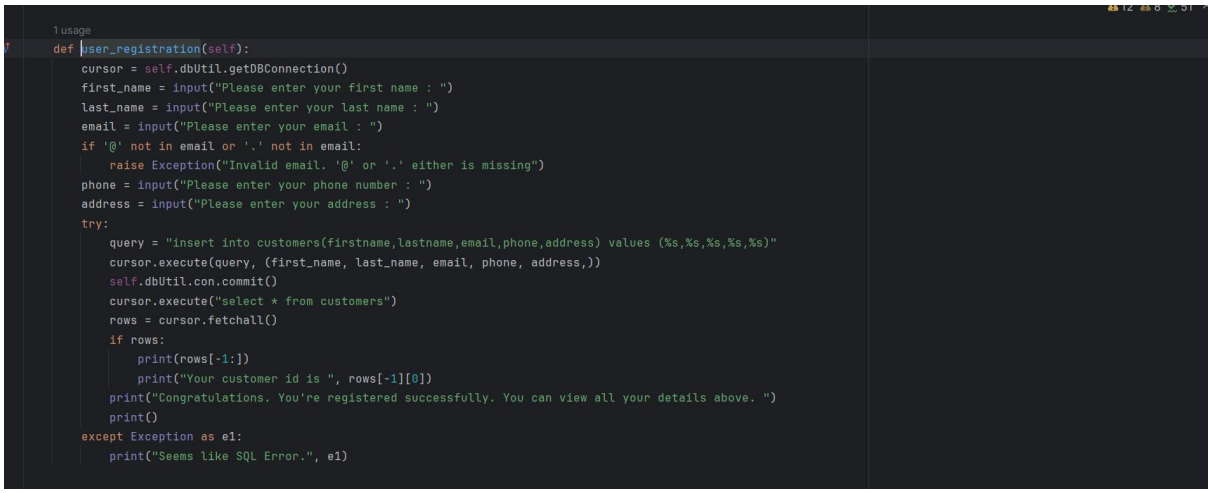
Shop › main() › while True

```python
                    self.process_payments()
                    print("We're heading you to main menu.")
                case 9:
                    self.search_products()
                    print()
                case 10:
                    print("Thanks for visiting us. See you anytime soon.")


dbutil = DBUtil()
techshop = TechShop(dbutil)
techshop.main()
```

1. Customer Registration:

Code:



```
1 usage
def user_registration(self):
    cursor = self.dbUtil.getDBConnection()
    first_name = input("Please enter your first name : ")
    last_name = input("Please enter your last name : ")
    email = input("Please enter your email : ")
    if '@' not in email or '.' not in email:
        raise Exception("Invalid email. '@' or '.' either is missing")
    phone = input("Please enter your phone number : ")
    address = input("Please enter your address : ")
    try:
        query = "insert into customers(firstname,lastname,email,phone,address) values (%s,%s,%s,%s,%s)"
        cursor.execute(query, (first_name, last_name, email, phone, address,))
        self.dbUtil.con.commit()
        cursor.execute("select * from customers")
        rows = cursor.fetchall()
        if rows:
            print(rows[-1:])
            print("Your customer id is ", rows[-1][0])
        print("Congratulations. You're registered successfully. You can view all your details above. ")
        print()
    except Exception as e1:
        print("Seems like SQL Error.", e1)
```

Output:

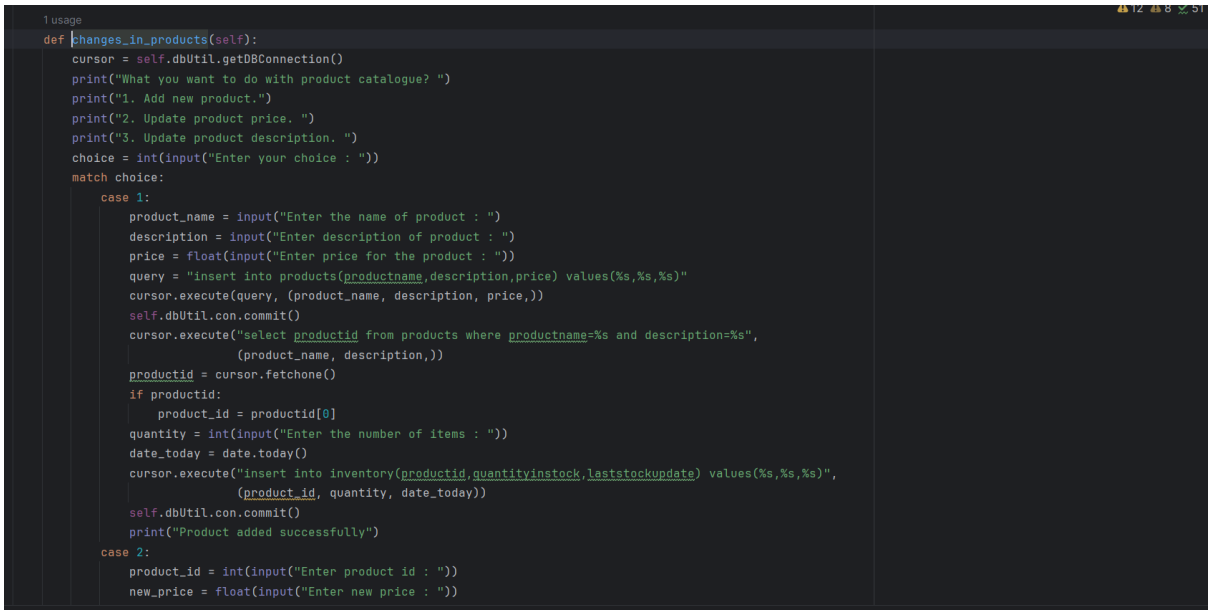
```
7. Customer Account Updates
8. Payment Processing
9. Product Search and Recommendations
10. Exit
Enter your choice here : 1
Please enter your first name : Somya
Please enter your last name : Kumar
Please enter your email : somya@outlook.com
Please enter your phone number : 9166894461
Please enter your address : Gannipur,MFP
[(17, 'Somya', 'Kumar', 'somya@outlook.com', '9166894461', 'Gannipur,MFP')]
Your customer id is  17
Congratulations. You're registered successfully. You can view all your details above.
```

Updated in database:

```
mysql> select * from customers;
+------------+-----------+-----------+----------------------------+------------+---------------------------------+
| customerid | FirstName | LastName  | Email                      | Phone      | Address                         |
+------------+-----------+-----------+----------------------------+------------+---------------------------------+
|          1 | John      | Doe       | john.doe@email.com         | 1234567890 | 123 Main Street, City           |
|          2 | Jane      | Smith     | jane.smith@email.com       | 9876543210 | 456 Oak Avenue, Town            |
|          3 | Bob       | Johnson   | bob.johnson@email.com      | 5551234567 | 789 Pine Road, Village          |
|          4 | Alice     | Brown     | alice.brown@email.com      | 1112223333 | 321 Elm Lane, Countryside       |
|          5 | Charlie   | Davis     | charlie.davis@email.com    | 4445556666 | 567 Maple Street, Suburb        |
|          6 | Emily     | White     | emily.white@email.com      | 7778889999 | 876 Cedar Road, Hamlet          |
|          7 | David     | Miller    | david.miller@email.com     | 9990001111 | 234 Birch Court, Riverside      |
|          8 | Grace     | Wilson    | grace.wilson@email.com     | 3334445555 | 432 Pine Avenue, Metropolis     |
|          9 | Henry     | Lee       | henry.lee@email.com        | 6667778888 | 789 Oak Drive, City             |
|         10 | Olivia    | Turner    | turner.olivia@email.com    | 2223334444 | 954 Maple Road, New York        |
|         11 | Sarthak   | Shandilya | sarthaksandilyakm@gmail.com | 7320047161 | H no 11,RDS college campus,Bihar |
|         12 | Aayushi   | Singh     | saayushi383@gmail.com      | 9213495612 | Gorakhpur                       |
|         13 | Sarthak   | Shandilya | sarthaksandilyakm@gmail.com | 9123402704 | RDS college campus,Muzaffarpur  |
|         14 | Mayank    | Kumar     | mayank@outlook.com         | 9430203375 | Proffessor colony, Aghoria chowk |
|         16 | Raj Kumar | Jha       | raj@outlook.com            | 9523785655 | Arariya,Bihar                   |
|         17 | Somya     | Kumar     | somya@outlook.com          | 9166894461 | Gannipur,MFP                    |
+------------+-----------+-----------+----------------------------+------------+---------------------------------+
16 rows in set (0.00 sec)
```

2. Update product info:

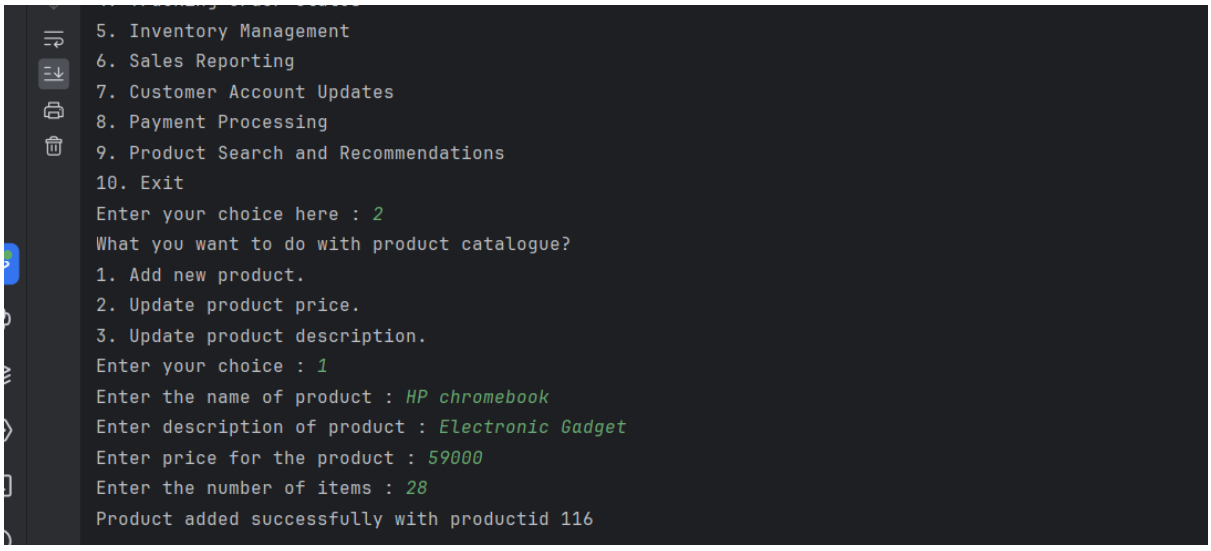
Code:



```
1 usage
def changes_in_products(self):
    cursor = self.dbUtil.getDBConnection()
    print("What you want to do with product catalogue? ")
    print("1. Add new product.")
    print("2. Update product price. ")
    print("3. Update product description. ")
    choice = int(input("Enter your choice : "))
    match choice:
        case 1:
            product_name = input("Enter the name of product : ")
            description = input("Enter description of product : ")
            price = float(input("Enter price for the product : "))
            query = "insert into products(productname,description,price) values(%s,%s,%s)"
            cursor.execute(query, (product_name, description, price,))
            self.dbUtil.con.commit()
            cursor.execute("select productid from products where productname=%s and description=%s",
                           (product_name, description,))
            productid = cursor.fetchone()
            if productid:
                product_id = productid[0]
            quantity = int(input("Enter the number of items : "))
            date_today = date.today()
            cursor.execute("insert into inventory(productid,quantityinstock,laststockupdate) values(%s,%s,%s)",
                           (product_id, quantity, date_today))
            self.dbUtil.con.commit()
            print("Product added successfully")
        case 2:
            product_id = int(input("Enter product id : "))
            new_price = float(input("Enter new price : "))
```

```
            self.dbUtil.con.commit()
            print("Product added successfully")
        case 2:
            product_id = int(input("Enter product id : "))
            new_price = float(input("Enter new price : "))
            cursor.execute("update products set price=%s where productid=%s", (new_price, product_id,))
            self.dbUtil.con.commit()
            print("Price updated successfully.")
        case 3:
            product_id = int(input("Enter product id : "))
            description = input("Enter new description of product : ")
            cursor.execute("update products set description=%s where productid=%s", (description, product_id))
            self.dbUtil.con.commit()
            print("Description updated successfully")
        case _:
            print("Invalid input.")
    print()
```

Output:



```
5. Inventory Management
6. Sales Reporting
7. Customer Account Updates
8. Payment Processing
9. Product Search and Recommendations
10. Exit
Enter your choice here : 2
What you want to do with product catalogue?
1. Add new product.
2. Update product price.
3. Update product description.
Enter your choice : 1
Enter the name of product : HP chromebook
Enter description of product : Electronic Gadget
Enter price for the product : 59000
Enter the number of items : 28
Product added successfully with productid 116
```

Updated in product and inventory database:

```
|       111 | EarBuds       | Audio                                       |    2200 |
|       113 | iPhone 14     | Brand new iPhone 14 with alot of features. |   70000 |
|       114 | Samsung A53   | Electronic Gadget                           |   25000 |
|       115 | HP Pavillion  | Electronic Gadget                           |   60000 |
|       116 | HP chromebook | Electronic Gadget                           |   59000 |
+-----------+---------------+---------------------------------------------+---------+
15 rows in set (0.00 sec)

mysql> select * from inventory;
+-------------+-----------+-----------------+-----------------+
| inventoryid | ProductID | QuantityInStock | LastStockUpdate |
+-------------+-----------+-----------------+-----------------+
|         201 |       101 |              50 | 2024-02-04      |
|         202 |       102 |              50 | 2024-01-11      |
|         203 |       103 |              10 | 2024-01-12      |
|         204 |       104 |              13 | 2024-01-13      |
|         205 |       105 |              28 | 2024-01-14      |
|         206 |       106 |               8 | 2024-01-15      |
|         207 |       107 |              25 | 2024-01-16      |
|         208 |       108 |              12 | 2024-01-17      |
|         209 |       109 |              40 | 2024-01-18      |
|         210 |       110 |              18 | 2024-01-19      |
|         211 |       113 |              15 | 2024-02-03      |
|         212 |       114 |              30 | 2024-02-04      |
|         213 |       115 |              35 | 2024-02-04      |
|         214 |       116 |              28 | 2024-02-04      |
+-------------+-----------+-----------------+-----------------+
14 rows in set (0.00 sec)
```

Updating price:

```
8. Payment Processing
9. Product Search and Recommendations
10. Exit
Enter your choice here : 2
What you want to do with product catalogue?
1. Add new product.
2. Update product price.
3. Update product description.
Enter your choice : 2
Enter product id : 115
Enter new price : 54000.00
Price updated successfully.
```

Reflected in database:

```
mysql> select * from products;
+-----------+---------------+---------------------------------------------+---------+
| productid | ProductName   | Description                                 | Price   |
+-----------+---------------+---------------------------------------------+---------+
|       101 | Laptop        | Electronic Gadget                           |    1320 |
|       102 | Smartphone    | Electronic Gadget                           | 769.989 |
|       103 | Headphones    | Audio                                       | 109.945 |
|       104 | Tablet        | Electronic Gadget                           | 549.989 |
|       105 | Smartwatch    | Electronic Gadget                           | 219.989 |
|       106 | Camera        | Camera                                      | 989.945 |
|       107 | Printer       | Input Output Devices                        | 142.989 |
|       108 | Speaker       | Audio                                       |  87.989 |
|       109 | Mouse         | Input Output Devices                        |  32.989 |
|       110 | Keyboard      | Input Output Devices                        | 164.989 |
|       111 | EarBuds       | Audio                                       |    2200 |
|       113 | iPhone 14     | Brand new iPhone 14 with alot of features. |   70000 |
|       114 | Samsung A53   | Electronic Gadget                           |   25000 |
|       115 | HP Pavillion  | Electronic Gadget                           |   54000 |
|       116 | HP chromebook | Electronic Gadget                           |   59000 |
+-----------+---------------+---------------------------------------------+---------+
15 rows in set (0.00 sec)
```

3. Placing new order:

Code:

```
def placing_order(self):
    cursor = self.dbUtil.getDBConnection()
    print("Please choose from the products below what you want to order : ")
    cursor.execute("select * from products")
    rows = cursor.fetchall()
    for row in rows:
        print(row)
    product_id = int(input("Please enter your product id here : "))
    quantity = int(input("Enter the number of items you want to order : "))
    new_customer = input("Are you a new customer? Enter yes or no : ")
    match new_customer:
        case "yes":
            first_name = input("First name : ")
            last_name = input("Last name : ")
            email = input("Email : ")
            try:
                if '@' in email and '.' in email:
                    pass
                else:
                    raise Exception("Incorrect input.")
            except Exception as e1:
                print("@ or . is missing. ", e1)
            phone = input("Phone : ")
            address = input("Address : ")
            q = "insert into customers(firstname,lastname,email,phone,address) values (%s,%s,%s,%s,%s)"
            cursor.execute(q, (first_name, last_name, email, phone, address,))
            self.dbUtil.con.commit()
            cursor.execute("select customerid from customers where firstname=%s and email=%s", (first_name, email,))
            customer = cursor.fetchone()
            if customer:
                customer_id = customer[0]
        case "no":
            customer_id = int(input("Enter your customer id : "))
        case _:
            print("Invalid input")
    cursor.execute("select price from products where productid=%s", (product_id,))
    amount = cursor.fetchone()
    if amount:
        total_amount = amount[0] * quantity
    date_today = date.today()
    placing_order = "insert into orders(customerid,orderdate,totalamount) values (%s,%s,%s)"
    cursor.execute(placing_order, (customer_id, date_today, total_amount,))
    self.dbUtil.con.commit()
    cursor.execute("select orderid from orders where customerid=%s and orderdate=%s", (customer_id, date_today,))
    orderid = cursor.fetchall()
    if orderid:
        order_id = orderid[-1][-1]
    orderdetailsupdate = "insert into orderdetails(orderid,productid,quantity) values(%s,%s,%s)"
    cursor.execute(orderdetailsupdate, (order_id, product_id, quantity,))
    self.dbUtil.con.commit()
    cursor.execute("select quantityinstock from inventory where productid=%s", (product_id,))
    quantity_in = cursor.fetchone()
    if quantity_in:
        quantity_in_stock = quantity_in[0]
    cursor.execute("update inventory set quantityinstock = %s where productid=%s",
                   (quantity_in_stock - quantity, product_id,))
    self.dbUtil.con.commit()
    print(f"Congratulations! Your order is successfully placed. Your order id is {order_id} and your total cost is {total_amount}")
    print()
```

Placing order for already existing customer:

```
8. Payment Processing
9. Product Search and Recommendations
10. Exit
Enter your choice here : 3
Please choose from the products below what you want to order :
(101, 'Laptop', 'Electronic Gadget', 1320.0)
(102, 'Smartphone', 'Electronic Gadget', 769.989)
(103, 'Headphones', 'Audio', 109.945)
(104, 'Tablet', 'Electronic Gadget', 549.989)
(105, 'Smartwatch', 'Electronic Gadget', 219.989)
(106, 'Camera', 'Camera', 989.945)
(107, 'Printer', 'Input Output Devices', 142.989)
(108, 'Speaker', 'Audio', 87.989)
(109, 'Mouse', 'Input Output Devices', 32.989)
(110, 'Keyboard', 'Input Output Devices', 164.989)
(111, 'EarBuds', 'Audio', 2200.0)
(113, 'iPhone 14', 'Brand new iPhone 14 with alot of features.', 70000.0)
(114, 'Samsung A53', 'Electronic Gadget', 25000.0)
(115, 'HP Pavillion', 'Electronic Gadget', 54000.0)
(116, 'HP chromebook', 'Electronic Gadget', 59000.0)
Please enter your product id here : 106
Enter the number of items you want to order : 2
Are you a new customer? Enter yes or no : no
Enter your customer id : 11
Congratulations! Your order is successfully placed. Your order id is 519 and your total cost is 1979.89

Order placed successfully.
```

Placing order for new customer:

```
9. Product Search and Recommendations
10. Exit
Enter your choice here : 3
Please choose from the products below what you want to order :
(101, 'Laptop', 'Electronic Gadget', 1320.0)
(102, 'Smartphone', 'Electronic Gadget', 769.989)
(103, 'Headphones', 'Audio', 109.945)
(104, 'Tablet', 'Electronic Gadget', 549.989)
(105, 'Smartwatch', 'Electronic Gadget', 219.989)
(106, 'Camera', 'Camera', 989.945)
(107, 'Printer', 'Input Output Devices', 142.989)
(108, 'Speaker', 'Audio', 87.989)
(109, 'Mouse', 'Input Output Devices', 32.989)
(110, 'Keyboard', 'Input Output Devices', 164.989)
(111, 'EarBuds', 'Audio', 2200.0)
(113, 'iPhone 14', 'Brand new iPhone 14 with alot of features.', 70000.0)
(114, 'Samsung A53', 'Electronic Gadget', 25000.0)
(115, 'HP Pavillion', 'Electronic Gadget', 54000.0)
(116, 'HP chromebook', 'Electronic Gadget', 59000.0)
Please enter your product id here : 115
Enter the number of items you want to order : 1
Are you a new customer? Enter yes or no : yes
First name : Sudhanshu
Last name : Malviya
Email : sudhanshu.malviya@outlook.com
Phone : 1234554322
Address : Aurangabad, Bihar
Congratulations! Your order is successfully placed. Your order id is 520 and your total cost is 54000.0
```

Updates reflecting in database:

```
mysql> select * from customers;
+------------+-----------+-----------+-----------------------------+------------+---------------------------------+
| customerid | FirstName | LastName  | Email                       | Phone      | Address                         |
+------------+-----------+-----------+-----------------------------+------------+---------------------------------+
|          1 | John      | Doe       | john.doe@email.com          | 1234567890 | 123 Main Street, City           |
|          2 | Jane      | Smith     | jane.smith@email.com        | 9876543210 | 456 Oak Avenue, Town            |
|          3 | Bob       | Johnson   | bob.johnson@email.com       | 5551234567 | 789 Pine Road, Village          |
|          4 | Alice     | Brown     | alice.brown@email.com       | 1112223333 | 321 Elm Lane, Countryside       |
|          5 | Charlie   | Davis     | charlie.davis@email.com     | 4445556666 | 567 Maple Street, Suburb        |
|          6 | Emily     | White     | emily.white@email.com       | 7778889999 | 876 Cedar Road, Hamlet          |
|          7 | David     | Miller    | david.miller@email.com      | 9990001111 | 234 Birch Court, Riverside      |
|          8 | Grace     | Wilson    | grace.wilson@email.com      | 3334445555 | 432 Pine Avenue, Metropolis     |
|          9 | Henry     | Lee       | henry.lee@email.com         | 6667778888 | 789 Oak Drive, City             |
|         10 | Olivia    | Turner    | turner.olivia@email.com     | 2223334444 | 954 Maple Road, New York        |
|         11 | Sarthak   | Shandilya | sarthaksandilyakm@gmail.com | 7320047161 | H no 11,RDS college campus,Bihar |
|         12 | Aayushi   | Singh     | saayushi383@gmail.com       | 9213495612 | Gorakhpur                       |
|         13 | Sarthak   | Shandilya | sarthaksandilyakm@gmail.com | 9123402704 | RDS college campus,Muzaffarpur  |
|         14 | Mayank    | Kumar     | mayank@outlook.com          | 9430203375 | Proffessor colony, Aghoria chowk |
|         16 | Raj Kumar | Jha       | raj@outlook.com             | 9523785655 | Arariya,Bihar                   |
|         17 | Somya     | Kumar     | somya@outlook.com           | 9166894461 | Gannipur,MFP                    |
|         18 | Sudhanshu | Malviya   | sudhanshu.malviya@outlook.com | 1234554322 | Aurangabad, Bihar               |
+------------+-----------+-----------+-----------------------------+------------+---------------------------------+
17 rows in set (0.00 sec)
```

```
mysql> select * from orders;
+----------+------------+------------+-------------+------------+
| orderid  | CustomerID | OrderDate  | TotalAmount | status     |
+----------+------------+------------+-------------+------------+
|      502 |          2 | 2024-01-11 |     1539.98 | Delivered  |
|      503 |          3 | 2024-01-12 |     329.835 | Delivered  |
|      504 |          4 | 2024-01-13 |     549.989 | Delivered  |
|      506 |          6 | 2024-01-15 |     989.945 | Shipped    |
|      507 |          7 | 2024-01-16 |     142.989 | Shipped    |
|      508 |          8 | 2024-01-17 |     175.978 | Shipped    |
|      509 |          9 | 2024-01-18 |      98.967 | Shipped    |
|      510 |         10 | 2024-01-19 |     164.989 | Shipped    |
|      511 |         11 | 2024-02-03 |     439.978 | Shipped    |
|      512 |         11 | 2024-02-03 |     659.967 | Shipped    |
|      513 |         11 | 2024-02-03 |     659.967 | Shipped    |
|      514 |         11 | 2024-02-03 |     175.978 | Processing |
|      515 |         11 | 2024-02-03 |        4400 | Processing |
|      516 |         11 | 2024-02-03 |     329.978 | Processing |
|      517 |         11 | 2024-02-03 |     549.989 | Processing |
|      518 |         11 | 2024-02-03 |     549.989 | Processing |
|      519 |         11 | 2024-02-04 |     1979.89 | NULL       |
|      520 |         18 | 2024-02-04 |       54000 | NULL       |
+----------+------------+------------+-------------+------------+
18 rows in set (0.01 sec)
```

Note: Order status we will be updating in further part of the project

4. Track order status

```
E:\HexawareAssignments\Python\TechShop\.venv\Scripts\python.exe E:\HexawareAssignments\Python\TechShop\app.py
Please select the choices from below :
1. Customer Registration.
2. Update Product Info.
3. Place New Order
4. Tracking Order Status
5. Inventory Management
6. Sales Reporting
7. Customer Account Updates
8. Payment Processing
9. Product Search and Recommendations
10. Exit
Enter your choice here : 4
Please enter your order id : 520
Your order with order id 520 is Shipped. Rest assured and contact us for further enquiry

We're heading you to main menu.
Please select the choices from below :
1. Customer Registration.
2. Update Product Info.
3. Place New Order
4. Tracking Order Status
5. Inventory Management
6. Sales Reporting
7. Customer Account Updates
8. Payment Processing
9. Product Search and Recommendations
10. Exit
Enter your choice here :
```

5. Inventory Management

```
2 usages
def manage_inventory(self):
    cursor = self.dbUtil.getDBConnection()
    print("Select from below what you want to do.")
    print("1. Add new product.")
    print("2. Update any product quantity stock.")
    print("3. Remove discontinued products.")
    choice = int(input("Enter your choice here : "))
    match choice:
        case 1:
            product_name = input("Enter product name : ")
            category = input("Enter category of product : ")
            price = float(input("Enter price of the product : "))
            cursor.execute("insert into products(productname, description, price) values (%s,%s,%s)",(product_name, category, price, ))
            self.dbUtil.con.commit()
            cursor.execute("select productid from products where productname=%s limit 1", (product_name,))
            product_ids = cursor.fetchone()
            if product_ids:
                product_id = product_ids[0]
            date_today = date.today()
            quantity = int(input("Enter the stock of product (quantity) : "))
            cursor.execute("insert into inventory(productid,quantityinstock,laststockupdate) values (%s,%s,%s)", (product_id, quantity, date_today, ))
            self.dbUtil.con.commit()
            cursor.execute("select inventoryid from inventory where productname=%s and laststockupdate=%s limit 1", (product_name, date_today, ))
            in_ids = cursor.fetchone()
            if in_ids:
                inventory_id = in_ids[0]
            print(f"Product is added to inventory and products successfully and your product id is {product_id} where inventory id is {inventory_id} .")
            print("Heading you to main menu.")
        case 2:
iceImpl › manage_inventory()
```

```
            print(f"Product is added to inventory and products successfully and your product id is {product_id} where inventory id is {inventory_id} .")
            print("Heading you to main menu.")
        case 2:
            product_id = int(input("Please provide product id whose quantity you want to update : "))
            cursor.execute("select inventoryid from inventory where productid=%s limit 1", (product_id, ))
            inventory_id = cursor.fetchone()[0]
            quantity = int(input("Enter new quantity : "))
            date_today = date.today()
            cursor.execute("update inventory set quantityinstock=%s, laststockupdate=%s where productid=%s", (quantity, date_today, product_id, ))
            self.dbUtil.con.commit()
            print(f"Your updates are implemented successfully for inventory id {inventory_id}")
        case 3:
            product_id = int(input("Please enter product id of the product discontinued : "))
            cursor.execute("update inventory set quantity=0, laststockupdate=%s where productid=%s", (date.today(), product_id, ))
            self.dbUtil.con.commit()
        case _:
            print("Invalid input.")
            print("Try again.")
            self.manage_inventory()
    print("Heading you to main menu. ")
    print()
```

Output:

```
Please select the choices from below :
1. Customer Registration.
2. Update Product Info.
3. Place New Order
4. Tracking Order Status
5. Inventory Management
6. Sales Reporting
7. Customer Account Updates
8. Payment Processing
9. Product Search and Recommendations
10. Exit
Enter your choice here : 5
Select from below what you want to do.
1. Add new product.
2. Update any product quantity stock.
3. Remove discontinued products.
Enter your choice here : 2
Please provide product id whose quantity you want to update : 110
Enter new quantity : 36
Your updates are implemented successfully for inventory id 210
Heading you to main menu.

We're heading you to main menu.
Please select the choices from below :
1. Customer Registration.
2. Update Product Info.
```

6. Sales reporting

Code:

```
1 usage
def report_sales(self):
    cursor = self.dbUtil.getDBConnection()
    cursor.execute("select sum(totalamount),orderdate from orders group by orderdate")
    rows = cursor.fetchall()
    for row in rows:
        print(row)
    print("These are amount sums according to order date.")
```

Output:

Note: Sales report is according to order date.

```
Enter your choice here : 6
(1539.97802734375, datetime.date(2024, 1, 11))
(329.8349914550781, datetime.date(2024, 1, 12))
(549.989013671875, datetime.date(2024, 1, 13))
(989.9450073242188, datetime.date(2024, 1, 15))
(142.989013671875, datetime.date(2024, 1, 16))
(175.97799682617188, datetime.date(2024, 1, 17))
(98.96699523925781, datetime.date(2024, 1, 18))
(164.989013671875, datetime.date(2024, 1, 19))
(7765.845977783203, datetime.date(2024, 2, 3))
(55979.89001464844, datetime.date(2024, 2, 4))
These are amount sums according to order date.
We're heading you to main menu.
```

7. Customer Account Updates:

Code:

```
1 usage
def customer_updates(self):
    customer_id = int(input("Please enter your customer id : "))
    print("Tell us what you want to do : ")
    print("1. Update email.")
    print("2. Update phone number.")
    print("3. Update address. ")
    cursor = self.dbUtil.getDBConnection()
    choice = int(input("Enter your choice here : "))
    match choice:
        case 1:
            email_id = input("Enter your new email address : ")
            cursor.execute("select email from customers where customerid=%s", (customer_id,))
            old_mails = cursor.fetchone()
            if old_mails:
                old_email = old_mails[0]
            cursor.execute("update customers set email=%s where customerid=%s", (email_id, customer_id,))
            self.dbUtil.con.commit()
            print(f"Your email is updated from {old_email} to {email_id} successfully.")
        case 2:
            phone_number = input("Enter your new phone number : ")
            cursor.execute("select phone from customers where customerid=%s", (customer_id,))
            old_numbers = cursor.fetchone()
            if old_numbers:
                old_number = old_numbers[0]
            cursor.execute("update customers set phone=%s where customerid=%s", (phone_number, customer_id,))
            self.dbUtil.con.commit()
            print(f"Your phone number is updated successfully from {old_number} to {phone_number}")
ServiceImpl › customer_updates()
```

```
            print(f"Your phone number is updated successfully from {old_number} to {phone_number}")
        case 3:
            address = input("Enter your new address : ")
            cursor.execute("select address from customers where customerid=%s", (customer_id,))
            adress = cursor.fetchone()
            if adress:
                old_address = adress[0]
            cursor.execute("update customers set address=%s where customerid=%s", (address, customer_id,))
            self.dbUtil.con.commit()
            print(f"Your address is updated successfully from {old_address} to {address}")
        case _:
            print("Invalid input.")
    print()
```

Output:

```
10. Exit
Enter your choice here : 7
Please enter your customer id : 2
Tell us what you want to do :
1. Update email.
2. Update phone number.
3. Update address.
Enter your choice here : 1
Enter your new email address : doe.john@yahoo.com
Your email is updated from jane.smith@email.com to doe.john@yahoo.com successfully.

Updates were successful.
We're heading you to main menu.
Please select the choices from below :
1. Customer Registration.
```

Updates updated in database:

```
mysql> select * from customers;
+------------+-----------+-----------+-------------------------------+------------+---------------------------------+
| customerid | FirstName | LastName  | Email                         | Phone      | Address                         |
+------------+-----------+-----------+-------------------------------+------------+---------------------------------+
|          1 | John      | Doe       | john.doe@email.com            | 1234567890 | 123 Main Street, City           |
|          2 | Jane      | Smith     | doe.john@yahoo.com            | 9876543210 | 456 Oak Avenue, Town            |
|          3 | Bob       | Johnson   | bob.johnson@email.com         | 5551234567 | 789 Pine Road, Village          |
|          4 | Alice     | Brown     | alice.brown@email.com         | 1112223333 | 321 Elm Lane, Countryside       |
|          5 | Charlie   | Davis     | charlie.davis@email.com       | 4445556666 | 567 Maple Street, Suburb        |
|          6 | Emily     | White     | emily.white@email.com         | 7778889999 | 876 Cedar Road, Hamlet          |
|          7 | David     | Miller    | david.miller@email.com        | 9990001111 | 234 Birch Court, Riverside      |
|          8 | Grace     | Wilson    | grace.wilson@email.com        | 3334445555 | 432 Pine Avenue, Metropolis     |
|          9 | Henry     | Lee       | henry.lee@email.com           | 6667778888 | 789 Oak Drive, City             |
|         10 | Olivia    | Turner    | turner.olivia@email.com       | 2223334444 | 954 Maple Road, New York        |
|         11 | Sarthak   | Shandilya | sarthaksandilyakm@gmail.com   | 7320047161 | H no 11,RDS college campus,Bihar |
|         12 | Aayushi   | Singh     | saayushi383@gmail.com         | 9213495612 | Gorakhpur                       |
|         13 | Sarthak   | Shandilya | sarthaksandilyakm@gmail.com   | 9123402704 | RDS college campus,Muzaffarpur  |
|         14 | Mayank    | Kumar     | mayank@outlook.com            | 9430203375 | Proffessor colony, Aghoria chowk |
|         16 | Raj Kumar | Jha       | raj@outlook.com               | 9523785655 | Arariya,Bihar                   |
|         17 | Somya     | Kumar     | somya@outlook.com             | 9166894461 | Gannipur,MFP                    |
|         18 | Sudhanshu | Malviya   | sudhanshu.malviya@outlook.com | 1234554322 | Aurangabad, Bihar               |
+------------+-----------+-----------+-------------------------------+------------+---------------------------------+
17 rows in set (0.01 sec)
```

Note: Email updated for customer with customer id 2.

8. Payment processing

Since payment method and every info was not available I left this section to update later on.

```
1 usage
def process_payments(self):
    print("The payment server is low at the time please come back later. Sorry for the inconvenience.")
```

9. Product Search and Recommendations:

Code:

```
1 usage
def search_products(self):
    cursor = self.dbUtil.getDBConnection()
    cursor.execute("select * from products")
    rows = cursor.fetchall()
    for row in rows:
        print(row)
    product_name = input("Please enter product name : ")
    cursor.execute("select * from products where productname=%s", (product_name,))
    product = cursor.fetchall()
    for p in product:
        print(p)
    cursor.execute("select description from products where productname = %s", (product_name,))
    des = cursor.fetchone()
    if des:
        description = des[0]
    cursor.execute("select * from products where description=%s", (description,))
    products = cursor.fetchall()
    print("Recommended Products .")
    for p in products:
        print(p)
    print("Go to place order section to order something.")
    print()
```

Output:

```
10. Exit
Enter your choice here : 9
(101, 'Laptop', 'Electronic Gadget', 1320.0)
(102, 'Smartphone', 'Electronic Gadget', 769.989)
(103, 'Headphones', 'Audio', 109.945)
(104, 'Tablet', 'Electronic Gadget', 549.989)
(105, 'Smartwatch', 'Electronic Gadget', 219.989)
(106, 'Camera', 'Camera', 989.945)
(107, 'Printer', 'Input Output Devices', 142.989)
(108, 'Speaker', 'Audio', 87.989)
(109, 'Mouse', 'Input Output Devices', 32.989)
(110, 'Keyboard', 'Input Output Devices', 164.989)
(111, 'EarBuds', 'Audio', 2200.0)
(113, 'iPhone 14', 'Brand new iPhone 14 with alot of features.', 70000.0)
(114, 'Samsung A53', 'Electronic Gadget', 25000.0)
(115, 'HP Pavillion', 'Electronic Gadget', 54000.0)
(116, 'HP chromebook', 'Electronic Gadget', 59000.0)
Please enter product name : HP Pavillion
(115, 'HP Pavillion', 'Electronic Gadget', 54000.0)
Recommended Products .
(101, 'Laptop', 'Electronic Gadget', 1320.0)
(102, 'Smartphone', 'Electronic Gadget', 769.989)
(104, 'Tablet', 'Electronic Gadget', 549.989)
(105, 'Smartwatch', 'Electronic Gadget', 219.989)
(114, 'Samsung A53', 'Electronic Gadget', 25000.0)
(115, 'HP Pavillion', 'Electronic Gadget', 54000.0)
(116, 'HP chromebook', 'Electronic Gadget', 59000.0)
Go to place order section to order something.
```

Abstract methods/Interface for TechShopService:

```python
from abc import *


2 usages
class TechShopService:
    @abstractmethod
    def user_registration(self):
        pass

    @abstractmethod
    def changes_in_products(self):
        pass

    @abstractmethod
    def placing_order(self):
        pass

    @abstractmethod
    def get_order_status(self):
        pass

    @abstractmethod
    def manage_inventory(self):
        pass

    @abstractmethod
    def report_sales(self):
        pass

    @abstractmethod
    def customer_updates(self):
```

```python
@abstractmethod
def customer_updates(self):
    pass

@abstractmethod
def process_payments(self):
    pass

@abstractmethod
def search_products(self):
    pass
```

TechShop Service Provider Implementation:

```python
from datetime import date
from abc import ABC

from TechShopService import TechShopService


2 usages
class TechShopServiceImpl(TechShopService, ABC):
    def __init__(self, dbUtil):
        self.dbUtil = dbUtil

    1 usage
    def customer_updates(self):
        customer_id = int(input("Please enter your customer id : "))
        print("Tell us what you want to do : ")
        print("1. Update email.")
        print("2. Update phone number.")
        print("3. Update address. ")
        cursor = self.dbUtil.getDBConnection()
        choice = int(input("Enter your choice here : "))
        match choice:
            case 1:
                email_id = input("Enter your new email address : ")
                cursor.execute("select email from customers where customerid=%s", (customer_id,))
                old_mails = cursor.fetchone()
                if old_mails:
                    old_email = old_mails[0]
                cursor.execute("update customers set email=%s where customerid=%s", (email_id, customer_id,))
                self.dbUtil.con.commit()
                print(f"Your email is updated from {old_email} to {email_id} successfully.")
            case 2:
```

```python
                    old_email = old_mails[0]
                cursor.execute("update customers set email=%s where customerid=%s", (email_id, customer_id,))
                self.dbUtil.con.commit()
                print(f"Your email is updated from {old_email} to {email_id} successfully.")
            case 2:
                phone_number = input("Enter your new phone number : ")
                cursor.execute("select phone from customers where customerid=%s", (customer_id,))
                old_numbers = cursor.fetchone()
                if old_numbers:
                    old_number = old_numbers[0]
                cursor.execute("update customers set phone=%s where customerid=%s", (phone_number, customer_id,))
                self.dbUtil.con.commit()
                print(f"Your phone number is updated successfully from {old_number} to {phone_number}")
            case 3:
                address = input("Enter your new address : ")
                cursor.execute("select address from customers where customerid=%s", (customer_id,))
                adress = cursor.fetchone()
                if adress:
                    old_address = adress[0]
                cursor.execute("update customers set address=%s where customerid=%s", (address, customer_id,))
                self.dbUtil.con.commit()
                print(f"Your address is updated successfully from {old_address} to {address}")
            case _:
                print("Invalid input.")
        print()

    1 usage
    def changes_in_products(self):
        cursor = self.dbUtil.getDBConnection()
        print("What you want to do with product catalogue? ")
        print("1. Add new product.")
```

```
1 usage
def changes_in_products(self):
    cursor = self.dbUtil.getDBConnection()
    print("What you want to do with product catalogue? ")
    print("1. Add new product.")
    print("2. Update product price. ")
    print("3. Update product description. ")
    choice = int(input("Enter your choice : "))
    match choice:
        case 1:
            product_name = input("Enter the name of product : ")
            description = input("Enter description of product : ")
            price = float(input("Enter price for the product : "))
            query = "insert into products(productname,description,price) values(%s,%s,%s)"
            cursor.execute(query, (product_name, description, price,))
            self.dbUtil.con.commit()
            cursor.execute("select productid from products where productname=%s and description=%s",
                           (product_name, description,))
            productid = cursor.fetchone()
            if productid:
                product_id = productid[0]
            quantity = int(input("Enter the number of items : "))
            date_today = date.today()
            cursor.execute("insert into inventory(productid,quantityinstock,laststockupdate) values(%s,%s,%s)",
                           (product_id, quantity, date_today))
            self.dbUtil.con.commit()
            print(f"Product added successfully with productid {product_id}")
        case 2:
            product_id = int(input("Enter product id : "))
```

```
                cursor.execute("insert into inventory(productid,quantityinstock,laststockupdate) values(%s,%s,%s)",
                               (product_id, quantity, date_today))
                self.dbUtil.con.commit()
                print(f"Product added successfully with productid {product_id}")
            case 2:
                product_id = int(input("Enter product id : "))
                new_price = float(input("Enter new price : "))
                cursor.execute("update products set price=%s where productid=%s", (new_price, product_id,))
                self.dbUtil.con.commit()
                print("Price updated successfully.")
            case 3:
                product_id = int(input("Enter product id : "))
                description = input("Enter new description of product : ")
                cursor.execute("update products set description=%s where productid=%s", (description, product_id))
                self.dbUtil.con.commit()
                print("Description updated successfully")
            case _:
                print("Invalid input.")
        print()
```

```
1 usage
def placing_order(self):
    cursor = self.dbUtil.getDBConnection()
    print("Please choose from the products below what you want to order : ")
    cursor.execute("select * from products")
    rows = cursor.fetchall()
    for row in rows:
        print(row)
    product_id = int(input("Please enter your product id here : "))
    quantity = int(input("Enter the number of items you want to order : "))
    new_customer = input("Are you a new customer? Enter yes or no : ")
    match new_customer:
        case "yes":
            first_name = input("First name : ")
            last_name = input("Last name : ")
            email = input("Email : ")
            try:
                if '@' in email and '.' in email:
                    pass
                else:
                    raise Exception("Incorrect input.")
            except Exception as e1:
                print("@ or . is missing. ", e1)
            phone = input("Phone : ")
            address = input("Address : ")
            q = "insert into customers(firstname,lastname,email,phone,address) values (%s,%s,%s,%s,%s)"
            cursor.execute(q, (first_name, last_name, email, phone, address,))
            self.dbUtil.con.commit()
            cursor.execute("select customerid from customers where firstname=%s and email=%s", (first_name, email,))
            customer = cursor.fetchone()
```

```
            customer = cursor.fetchone()
            if customer:
                customer_id = customer[0]
        case "no":
            customer_id = int(input("Enter your customer id : "))
        case _:
            print("Invalid input")
    cursor.execute("select price from products where productid=%s", (product_id,))
    amount = cursor.fetchone()
    if amount:
        total_amount = amount[0] * quantity
    date_today = date.today()
    placing_order = "insert into orders(customerid,orderdate,totalamount) values (%s,%s,%s)"
    cursor.execute(placing_order, (customer_id, date_today, total_amount,))
    self.dbUtil.con.commit()
    cursor.execute("select orderid from orders where customerid=%s and orderdate=%s", (customer_id, date_today,))
    orderid = cursor.fetchall()
    if orderid:
        order_id = orderid[-1][-1]
    orderdetailsupdate = "insert into orderdetails(orderid,productid,quantity) values(%s,%s,%s)"
    cursor.execute(orderdetailsupdate, (order_id, product_id, quantity,))
    self.dbUtil.con.commit()
    cursor.execute("select quantityinstock from inventory where productid=%s", (product_id,))
    quantity_in = cursor.fetchone()
    if quantity_in:
        quantity_in_stock = quantity_in[0]
    cursor.execute("update inventory set quantityinstock = %s where productid=%s",
                   (quantity_in_stock - quantity, product_id,))
    self.dbUtil.con.commit()
    print(f"Congratulations! Your order is successfully placed. Your order id is {order_id} and your total cost is {total_amount}")
    print()
```

```
2 usages
def get_order_status(self):
    order_id = int(input("Please enter your order id : "))
    cursor = self.dbUtil.getDBConnection()
    cursor.execute("select status from orders where orderid=%s", (order_id,))
    orders_result = cursor.fetchone()
    if orders_result is not None:
        status = orders_result[0]
    else:
        print("Incorrect order id please try again")
        self.get_order_status()
    print(f"Your order with order id {order_id} is {status}. Rest assured and contact us for further enquiry")
    print()
```

```
2 usages                                                                    12  8
def manage_inventory(self):
    cursor = self.dbUtil.getDBConnection()
    print("Select from below what you want to do.")
    print("1. Add new product.")
    print("2. Update any product quantity stock.")
    print("3. Remove discontinued products.")
    choice = int(input("Enter your choice here : "))
    match choice:
        case 1:
            product_name = input("Enter product name : ")
            category = input("Enter category of product : ")
            price = float(input("Enter price of the product : "))
            cursor.execute("insert into products(productname, description, price) values (%s,%s,%s)",(product_name, category, price, ))
            self.dbUtil.con.commit()
            cursor.execute("select productid from products where productname=%s limit 1", (product_name,))
            product_ids = cursor.fetchone()
            if product_ids:
                product_id = product_ids[0]
            date_today = date.today()
            quantity = int(input("Enter the stock of product (quantity) : "))
            cursor.execute("insert into inventory(productid,quantityinstock,laststockupdate) values (%s,%s,%s)", (product_id, quantity, date_today, ))
            self.dbUtil.con.commit()
            cursor.execute("select inventoryid from inventory where productname=%s and laststockupdate=%s limit 1", (product_name, date_today, ))
            in_ids = cursor.fetchone()
            if in_ids:
                inventory_id = in_ids[0]
            print(f"Product is added to inventory and products successfully and your product id is {product_id} where inventory id is {inventory_id} .")
            print("Heading you to main menu.")
        case 2:
            product_id = int(input("Please provide product id whose quantity you want to update : "))
```

```
            print("Heading you to main menu.")
        case 2:
            product_id = int(input("Please provide product id whose quantity you want to update : "))
            cursor.execute("select inventoryid from inventory where productid=%s limit 1", (product_id, ))
            inventory_id = cursor.fetchone()[0]
            quantity = int(input("Enter new quantity : "))
            date_today = date.today()
            cursor.execute("update inventory set quantityinstock=%s, laststockupdate=%s where productid=%s", (quantity, date_today, product_id, ))
            self.dbUtil.con.commit()
            print(f"Your updates are implemented successfully for inventory id {inventory_id}")
        case 3:
            product_id = int(input("Please enter product id of the product discontinued : "))
            cursor.execute("update inventory set quantity=0, laststockupdate=%s where productid=%s", (date.today(), product_id, ))
            self.dbUtil.con.commit()
        case _:
            print("Invalid input.")
            print("Try again.")
            self.manage_inventory()
    print("Heading you to main menu. ")
    print()
```

```
def report_sales(self):
    cursor = self.dbUtil.getDBConnection()
    cursor.execute("select sum(totalamount),orderdate from orders group by orderdate")
    rows = cursor.fetchall()
    for row in rows:
        print(row)
    print("These are amount sums according to order date.")

def user_registration(self):
    cursor = self.dbUtil.getDBConnection()
    first_name = input("Please enter your first name : ")
    last_name = input("Please enter your last name : ")
    email = input("Please enter your email : ")
    if '@' not in email or '.' not in email:
        raise Exception("Invalid email. '@' or '.' either is missing")
    phone = input("Please enter your phone number : ")
    address = input("Please enter your address : ")
    try:
        query = "insert into customers(firstname,lastname,email,phone,address) values (%s,%s,%s,%s,%s)"
        cursor.execute(query, (first_name, last_name, email, phone, address,))
        self.dbUtil.con.commit()
        cursor.execute("select * from customers")
        rows = cursor.fetchall()
        if rows:
            print(rows[-1:])
            print("Your customer id is ", rows[-1][0])
        print("Congratulations. You're registered successfully. You can view all your details above. ")
        print()
    except Exception as e1:
        print("Seems like SQL Error.", e1)
```

```
1 usage
def process_payments(self):
    print("The payment server is low at the time please come back later. Sorry for the inconvenience.")

1 usage
def search_products(self):
    cursor = self.dbUtil.getDBConnection()
    cursor.execute("select * from products")
    rows = cursor.fetchall()
    for row in rows:
        print(row)
    product_name = input("Please enter product name : ")
    cursor.execute("select * from products where productname=%s", (product_name,))
    product = cursor.fetchall()
    for p in product:
        print(p)
    cursor.execute("select description from products where productname = %s", (product_name,))
    des = cursor.fetchone()
    if des:
        description = des[0]
    cursor.execute("select * from products where description=%s", (description,))
    products = cursor.fetchall()
    print("Recommended Products .")
    for p in products:
        print(p)
    print("Go to place order section to order something.")
    print()
```

Customer Class Implementation:

```
2 usages
class InvalidDataException(Exception):
    def __init__(self, message):
        self.message = message


2 usages
class Customers:
    def __init__(self, customerID: int, firstName: str, lastName: str, email: str, phone: str, address: str):
        self.customerID = customerID
        self.firstName = firstName
        self.lastName = lastName
        self.email = email
        self.phone = phone
        self.address = address

    1 usage
    @property
    def getCustomerId(self):
        return self.customerID

    @getCustomerId.setter
    def setCustomerId(self, cid):
        self.customerID = cid

    1 usage
    @property
    def getName(self):
        return self.firstName + self.lastName

    @getName.setter
```

```
    @getName.setter
    def setName(self, fName, lName):
        self.firstName = fName
        self.lastName = lName

    1 usage
    @property
    def getEmail(self):
        return self.email

    @getEmail.setter
    def setEmail(self, email):
        try:
            if '@' in email:
                self.email = email
            else:
                raise InvalidDataException("Invalid email")
        except InvalidDataException as idv:
            print("@ is missing in the input provided. ", idv)
```

```
def CalculateTotalOrders(self):
    pass

def GetCustomerDetails(self):
    print("Customer ID = ", self.customerID)
    print("Customer Name = ", self.firstName, self.lastName)
    print("Customer Email = ", self.email)
    print("Customer Phone = ", self.phone)
    print("Customer Address = ", self.address)

def UpdateCustomerInfo(self, email):
    self.email = email
```

Products Class Implementation:

```
4 usages
class Products:
    def __init__(self, productID: int, productName, description, price: float):
        self.productID = productID
        self.productName = productName
        self.description = description
        self.price = price

    1 usage
    @property
    def getProductID(self):
        return self.productID

    1 usage
    @property
    def getProductName(self):
        return self.productName

    1 usage
    @property
    def getProductDescription(self):
        return self.description

    1 usage
    @property
    def getPrice(self):
        return self.price

    @getProductID.setter
    def setProductId(self,pid):
        self.productID = pid
s › UpdateProductInfo() › try › else
```

```
    @getProductID.setter
    def setProductId(self,pid):
        self.productID = pid

    @getProductName.setter
    def setProductName(self,name):
        self.productName = name

    @getProductDescription.setter
    def setProductDescription(self,desc):
        self.description = desc

    @getPrice.setter
    def setPrice(self, price):
        try:
            if price >= 0:
                self.price = price
            else:
                raise Exception("Price Can't be negative")
        except Exception as e1:
            print("Prices can't be negative please enter a positive value", e1)

    def GetProductDetails(self):
        print("Product ID = ", self.productID)
        print("Product Name = ", self.productName)
        print("Description = ", self.description)
        print("Price = ", self.price)
```

Orders Class Implementation:

```python
import datetime

from Customers import Customers


2 usages
class Orders(Customers):
    def __init__(self, orderID: int, customer, orderDate: datetime, totalAmount: float):
        self.orderID = orderID
        super().__init__(customer.customerID, customer.firstName, customer.lastName, customer.email, customer.phone, customer.address)
        self.orderDate = datetime.datetime.strptime(orderDate, __format: "%Y-%m-%d").date()
        self.totalAmount = totalAmount
        self.status = "Processing"

    1 usage
    @property
    def getOrderid(self):
        return self.orderID

    1 usage
    @property
    def getOrderDate(self):
        return self.orderDate

    1 usage
    @property
    def getTotalAmount(self):
        return self.totalAmount
```

```python
    @property
    def getStatus(self):
        return self.status

    @getOrderid.setter
    def setOrderid(self,oid):
        self.orderID = oid

    @getOrderDate.setter
    def setOrderDate(self,date):
        d = datetime.datetime.strptime(date, __format: "%Y-%m-%d")
        self.orderDate = d

    @getTotalAmount.setter
    def setTotalAmount(self,am):
        try:
            if am >= 0:
                self.totalAmount = am
            else:
                raise ValueError("Amount can't be negative")
        except ValueError as v1:
            print("Please enter valid amount. ", v1)

    @getStatus.setter
    def setStatus(self, status):
        self.status = status

    def CalculateTotalAmount(self):
        return self.totalAmount
```

OrderDetails Class Implementation:

```python
import Customers
from Orders import Orders
from Products import Products


class OrderDetails(Orders, Products):
    def __init__(self, orderDetailId: int, orders, products, quantity: int):
        self.orderDetailId = orderDetailId
        self.order = orders
        self.product = products
        self.quantity = quantity

    1 usage
    @property
    def getOrderDetailsId(self):
        return self.orderDetailId

    @getOrderDetailsId.setter
    def setOrderDetailsId(self,id):
        self.orderDetailId = id

    1 usage
    @property
    def getQuantity(self):
        return self.quantity
```

```python
@getQuantity.setter
def setQuantity(self,q):
    try:
        if q >= 0:
            self.quantity = q
        else:
            raise ValueError("Quantity must be negative")
    except ValueError as v1:
        print("Invalid Quantity. ", v1)


def CalculateSubTotal(self):
    totalAmount = self.order.totalAmount
    return totalAmount

def GetOrderDetailInfo(self):
    print(f"OrderDetail ID = {self.orderDetailId} ")
    print(f"Product Name = {self.product.productName}")
    print(f"Quantity = {self.quantity}")

def UpdateQuantity(self, newQuantity):
    try:
        if newQuantity >= 0:
            self.quantity = newQuantity
        else:
            raise ValueError("Quantity must be positive")
    except ValueError as e1:
        print("Sorry! You must enter a valid quantity.",e1)
```

Inventory Class Implementation:

```python
from Products import Products


class Inventory(Products):
    def __init__(self, inventoryID: int, product, quantityInStock: int, lastStockUpdate: int):
        self.inventoryID = inventoryID
        super().__init__(product.productID, product.productName, product.description,product.price)
        self.quantityInStock = quantityInStock
        self.lastStockUpdate = lastStockUpdate

    1 usage
    @property
    def getQuantityInStock(self):
        return self.quantityInStock

    @getQuantityInStock.setter
    def getQuantityInStock(self,quantity):
        self.quantityInStock = quantity

    def GetProduct(self):
        print(f"Inventory Id = {self.inventoryID}")
        print(f"Product Id = {self.productID}")
        print(f"Product Name = {self.productName}")
        print(f"Product Description = {self.description}")
        print(f"Product Price = {self.price}")
```

```python
def GetQuantityInStock(self):
    return self.quantityInStock

def AddToInventory(self, quantity):
    self.quantityInStock += quantity

def RemoveFromInventory(self, quantity):
    try:
        if quantity <= self.quantityInStock:
            self.quantityInStock -= quantity
            self.lastStockUpdate = self.quantityInStock
        else:
            raise ValueError("Quantity is greater than available quantity.")
    except ValueError as v1:
        print("Invalid quantity. ",v1)

def UpdateStockQuantity(self, newquantity):
    try:
        if newquantity >= 0:
            self.quantityInStock += newquantity
            self.lastStockUpdate = self.quantityInStock
        else:
            raise ValueError("Quanitity can't be negative.")
    except ValueError as v1:
        print("Please enter valid quantity.",v1)
```

```python
    def IsProductAvailable(self, quantity):
        if quantity >= self.quantityInStock:
            print(f"Yes! The product {self.productName} is available.")
        else:
            print("Sorry! The product is not available right now.")

    def GetInventoryValue(self):
        pass

    def ListLowStockProducts(self, threshold):
        if self.quantityInStock < threshold:
            print(f"Low stock for {self.productName}")
        else:
            print("The stock is available")

    def ListOutOfStockProducts(self):
        pass

    def ListAllProducts(self):
        pass
```