ASSIGNMENT 1

TECHSHOP

AKILESH K

```
class Customers:
    def __init__(self, customer_id, first_name, last_name, email, phone, address):
        self.CustomerID = customer_id
        self.FirstName = first_name
        self.LastName = last_name
        self.Email = email
        self.Phone = phone
        self.Address = address
        self.orders = []

    def calculate_total_orders(self):
        return len(self.orders)

    def get_customer_details(self):
        print("Customer ID:", self.CustomerID)
        print("Name:", self.FirstName, self.LastName)
        print("Email:", self.Email)
        print("Phone:", self.Phone)
        print("Address:", self.Address)
        print("Total Orders:", self.calculate_total_orders())

    def update_customer_info(self, new_email=None, new_phone=None, new_address=None):
        if new_email:
            self.Email = new_email
        if new_phone:
            self.Phone = new_phone
```

```
        if new_address:
            self.Address = new_address
        print("Customer information updated successfully.")



class Product:
    def __init__(self, product_id, name, price):
        self.ProductID = product_id
        self.Name = name
        self.Price = price

class Order:
    def __init__(self, order_id, customer, products):
        self.OrderID = order_id
        self.Customer = customer
        self.Products = products

    def calculate_order_total(self):
        return sum(product.Price for product in self.Products)



class Products:
    def __init__(self, product_id, product_name, description, price, stock):
        self.ProductID = product_id
        self.ProductName = product_name
        self.Description = description
        self.Price = price
        self.Stock = stock

    def get_product_details(self):
        print("Product ID:", self.ProductID)
```

```
        print("Product Name:", self.ProductName)
        print("Description:", self.Description)
        print("Price:", self.Price)
        print("Stock:", self.Stock)

    def update_product_info(self, new_price=None, new_description=None):
        if new_price:
            self.Price = new_price
        if new_description:
            self.Description = new_description
        print("Product information updated successfully.")

    def is_product_in_stock(self):
        return self.Stock > 0




from datetime import datetime

class Orders:
    def __init__(self, order_id, customer, order_date=None):
        self.OrderID = order_id
        self.Customer = customer  # Using composition to reference the Customer
        self.OrderDate = order_date or datetime.now()
        self.TotalAmount = 0
        self.order_details = []

    def calculate_total_amount(self):
        self.TotalAmount = sum(order_detail.calculate_subtotal() for order_detail in self.order_details)
        return self.TotalAmount
```

```
    def get_order_details(self):
        print("Order ID:", self.OrderID)
        print("Customer:", self.Customer.FirstName, self.Customer.LastName)
        print("Order Date:", self.OrderDate)
        print("Total Amount:", self.TotalAmount)
        for order_detail in self.order_details:
            order_detail.get_order_detail_info()

    def update_order_status(self, new_status):
        pass

    def cancel_order(self):
        pass

class OrderDetails:
    def __init__(self, order_detail_id, order, product, quantity):
        self.OrderDetailID = order_detail_id
        self.Order = order  # Using composition to reference the Order
        self.Product = product
        self.Quantity = quantity

    def calculate_subtotal(self):
        return self.Product.Price * self.Quantity

    def get_order_detail_info(self):
        print("Order Detail ID:", self.OrderDetailID)
        print("Product:", self.Product.ProductName)
        print("Quantity:", self.Quantity)
        print("Subtotal:", self.calculate_subtotal())

    def update_quantity(self, new_quantity):
```

```
        self.Quantity = new_quantity

    def add_discount(self, discount_amount):
        pass

class Inventory:
    def __init__(self, inventory_id, product, quantity_in_stock, last_stock_update=None):
        self.InventoryID = inventory_id
        self.Product = product
        self.QuantityInStock = quantity_in_stock
        self.LastStockUpdate = last_stock_update or datetime.now()

    def get_product(self):
        return self.Product

    def get_quantity_in_stock(self):
        return self.QuantityInStock

    def add_to_inventory(self, quantity):
        self.QuantityInStock += quantity
        self.LastStockUpdate = datetime.now()

    def remove_from_inventory(self, quantity):
        self.QuantityInStock -= quantity
        self.LastStockUpdate = datetime.now()

    def update_stock_quantity(self, new_quantity):
        self.QuantityInStock = new_quantity
        self.LastStockUpdate = datetime.now()

    def is_product_available(self, quantity_to_check):
```

```
        return self.QuantityInStock >= quantity_to_check

    def get_inventory_value(self):
        return self.Product.Price * self.QuantityInStock

    def list_low_stock_products(self, threshold):
        if self.QuantityInStock < threshold:
            print(f"Low stock product: {self.Product.ProductName}, Quantity: {self.QuantityInStock}")

    def list_out_of_stock_products(self):
        if self.QuantityInStock == 0:
            print(f"Out of stock product: {self.Product.ProductName}")

    def list_all_products(self):
        print(f"Product: {self.Product.ProductName}, Quantity: {self.QuantityInStock}")
```

Task 5

```
class InvalidDataException(Exception):
    pass
def validate_email(email):
    if not is_valid_email(email):
        raise InvalidDataException("Invalid email format.")

class InsufficientStockException(Exception):
    pass
def process_order(product_id, quantity):
    available_stock = get_available_stock(product_id)
    if quantity > available_stock:
        raise InsufficientStockException("Quantity exceeds available stock.")

class IncompleteOrderException(Exception):
```

```
    pass
def process_order(order_details):
    if 'product_reference' not in order_details:
        raise IncompleteOrderException("Order detail lacks product reference.")



def process_payment(order_id):
    try:
        make_payment(order_id)
    except PaymentFailedException as e:
        handle_payment_failure(e)

def write_to_log(log_entry):
    try:
        with open('log.txt', 'a') as log_file:
            log_file.write(log_entry + '\n')
    except IOError as e:
        handle_log_error(e)



def execute_sql_query(query):
    try:
        result = execute_query(query)
        return result
    except SqlException as e:
        handle_database_error(e)

def update_order(order_id, new_data):
    try:
        perform_update(order_id, new_data)
    except ConcurrencyException as e:
```

```
        handle_concurrency_error(e)



class AuthenticationException(Exception):
    pass
class AuthorizationException(Exception):
    pass
def check_access(user):
    if not user.is_authenticated:
        raise AuthenticationException("User not authenticated.")
    if not user.has_permission():
        raise AuthorizationException("Insufficient privileges.")
```

Task 7

```
def main_menu():
    while True:
        print("\nMain Menu:")
        print("1. Add New Customer")
        print("2. Make Order")
        print("3. Update Product Price")
        print("4. Get Order Details")
        print("5. Update Inventory")
        print("6. Get Order Summary")
        print("7. Update Customer Phone")
        print("8. Get Products by Description")
        print("9. Exit")

        choice = input("Enter your choice (1-9): ")

        if choice == '1':
```

```
            add_new_customer()
        elif choice == '2':
            make_order()
        elif choice == '3':
            update_product()
        elif choice == '4':
            get_order_details()
        elif choice == '5':
            update_inventory()
        elif choice == '6':
            get_order_summary()
        elif choice == '7':
            update_customer()
        elif choice == '8':
            get_desc()
        elif choice == '9':
            print("Exiting the program. Goodbye!")
            break
        else:
            print("Invalid choice. Please enter a number between 1 and 9.")

if __name__ == "__main__":
    main_menu()

from datetime import datetime
import mysql.connector
from mysql.connector import Error
# Database connection
def create_connection():
    try:
        connection = mysql.connector.connect(
```

```
        host="localhost",
        user="root",
        password="root",
        port='3306',
        database="techshop"
    )
    return connection
except Error as e:
    print(f"Error connecting to the database: {e}")
    return None
```

1.

```
c = 24

def generate_stno():
    global c
    c+= 1
    return c

def add_new_customer():
    connection = create_connection()
    if connection:
        try:
            customerid=generate_stno()
            firstname = input("Enter first name: ")
            lastname = input("Enter latname: ")
            email = input("Enter email: ")
            phone = input("Enter phone number: ")
            address = input("Enter address: ")
```

```
        cursor = connection.cursor()

        cursor.execute("INSERT INTO customers(customerid,firstname,lastname,email,phone,address)
VALUES (%s, %s, %s, %s,%s, %s)",

                (customerid,firstname,lastname,email,phone,address))

        connection.commit()


        print("customer details recorded successfully!")


    except (Error, ValueError) as e:

        print(f"Error recording customer: {e}")

    finally:

        connection.close()
```

```
PS C:\Users\Akilesh K\OneDrive\Documents\python class
hallenge\petpals'; & 'C:\Users\Akilesh K\AppData\Loca
-2023.22.1\pythonFiles\lib\python\debugpy\adapter/../
sign\techshop\tech.py'
Enter first name: aki
Enter latname: k
Enter email: aki@gmail
Enter phone number: 480340
Enter address: 4th cross
customer details recorded successfully!
```

```
mysql> select * from customers;
+------------+-----------+----------+-------------------------+----------------+---------------------------+
| customerid | firstname | lastname | email                   | phone          | address                   |
+------------+-----------+----------+-------------------------+----------------+---------------------------+
|          2 | Benjamin  | May      | benjamin@icloud.net     | 1-573-762-5734 | Ap #774-4948 Hendrerit Av. |
|          5 | Aiko      | Barry    | aiko@outlook.com        | 1-344-918-8823 | 2780 Non, Street          |
|          7 | Burton    | Hicks    | burton2976@protonmail.ca | (889) 371-0537 | 924-9041 Nullam Rd.       |
|          9 | Lani      | Moon     | lani6109@protonmail.com | (225) 833-2432 | 743-1961 Sapien Road      |
|         10 | Jonah     | Boyle    | jonah2869@yahoo.org     | (555) 377-2867 | st george street,12       |
|         25 | aki       | k        | aki@gmail               | 480340         | 4th cross                 |
+------------+-----------+----------+-------------------------+----------------+---------------------------+
6 rows in set (0.00 sec)
```

2.

```
def update_product():
    connection = create_connection()
    if connection:
        try:
            cursor = connection.cursor()
            productid=int(input("enter productid:"))
            new_price=int(input("enter new price:"))
            update_query = ("UPDATE products SET price = %s WHERE productid = %s")
            cursor.execute(update_query, (new_price, productid))

            connection.commit()
            print(f"Updated value in the database.")

        except Error as e:
            print(f"Error updating value in the table: {e}")

        finally:
            connection.close()
```

```
PS C:\Users\Akilesh K\OneDrive\Documents\pyth
hallenge\petpals'; & 'C:\Users\Akilesh K\AppD
-2023.22.1\pythonFiles\lib\python\debugpy\ada
sign\techshop\tech.py'
enter productid:32
enter new price:3000
Updated value in the database.
```

```
mysql> select * from products;
+-----------+-------------+-------------+-------+
| productid | productname | description | price |
+-----------+-------------+-------------+-------+
|        23 | keyboard    | gaming      |  1100 |
|        32 | imac        | pc          |  3000 |
|        44 | iphone      | red         |   275 |
|        65 | earpods     | white       |    44 |
|        73 | smartwatch  | esim        |   330 |
+-----------+-------------+-------------+-------+
5 rows in set (0.00 sec)
```

3.

```
t_w = 1020

def generate_tw_number():
    global t_w
    t_w += 1
    return t_w
w = 1025

def generate_o_number():
    global w
    w += 1
    return w

from functools import reduce
import operator

def make_order():
    connection = create_connection()
    if connection:
        try:
```

```
        cursor = connection.cursor()
        customerid=int(input("enter customer id:"))
        productid=int(input("enter order id:"))
        quantity=int(input("enter quantity:"))
        orderid=generate_tw_number()
        orderdeid=generate_o_number()
        t_date = datetime.now().strftime("%Y-%m-%d")
        select_query=("SELECT price FROM products where productid= %s")
        cursor.execute(select_query,(productid,))
        amount = cursor.fetchone()
        total=reduce(operator.__mul__, amount,quantity)



        cursor.execute("INSERT INTO orders (orderid,customerid,orderdate,totalamount) VALUES
(%s,%s, %s, %s)",
                    (orderid,customerid,t_date,total))


        cursor.execute("INSERT INTO orderdetails (orderdetailsid,orderid,productid,quantity) VALUES
(%s,%s, %s, %s)",
                    (orderdeid,orderid,productid,quantity))


        connection.commit()
        print(f"Updated value in the database.")


    except Error as e:
        print(f"Error updating value in the table: {e}")


    finally:
        connection.close()
```

```
PS C:\Users\Akilesh K\OneDrive\Documents\python class\coding challer
hallenge\petpals'; & 'C:\Users\Akilesh K\AppData\Local\Programs\Pyth
-2023.22.1\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\laur
sign\techshop\tech.py'
enter customer id:25
enter order id:32
enter quantity:4
Updated value in the database.
```

```
mysql> select * from orderdetails;
+----------------+---------+-----------+----------+
| orderdetailsid | orderid | productid | quantity |
+----------------+---------+-----------+----------+
|            532 |     742 |        73 |       22 |
|            625 |     638 |        44 |        2 |
|            807 |     524 |        65 |        8 |
|            837 |     331 |        32 |       10 |
|            972 |     937 |        23 |        5 |
|           1026 |    1021 |        32 |        4 |
+----------------+---------+-----------+----------+
6 rows in set (0.00 sec)

mysql> select * from orders;
+---------+------------+------------+-------------+
| orderid | customerid | orderdate  | totalamount |
+---------+------------+------------+-------------+
|     331 |          2 | 2023-07-21 |         330 |
|     524 |          9 | 2023-12-05 |         352 |
|     638 |         10 | 2023-11-30 |         550 |
|     742 |          5 | 2023-11-13 |        7260 |
|     937 |          9 | 2023-09-26 |        5500 |
|    1021 |         25 | 2023-12-25 |       12000 |
+---------+------------+------------+-------------+
6 rows in set (0.00 sec)
```

4.

```
def get_order_details():
    connection = create_connection()
    if connection:
        try:
            cursor = connection.cursor()
            customer_id=int(input("enter customer number:"))
```

```
        select_query=("SELECT * FROM orders where customerid= %s")
        cursor.execute(select_query,(customer_id,))
        cus = cursor.fetchall()

        print("transactions:")

        print(cus)

    except Error as e:
        print(f"Error retrieving listings: {e}")
    finally:
        connection.close()
```

```
enter customer number:25
transactions:
[(1021, 25, datetime.date(2023, 12, 25), 12000.0)]
PS C:\Users\Akilesh K\OneDrive\Documents\python class
```

5.

```
def update_inventory():
    connection = create_connection()
    if connection:
        try:
        cursor = connection.cursor()
        productid=int(input("enter product id:"))
        new_quan=input("enter the new quantity:")
        update_query = ("UPDATE inventory SET quantityinstock = %s WHERE productid = %s")
        cursor.execute(update_query, (new_quan, productid))

        connection.commit()
```

```
        print(f"Updated value in the database.")


    except Error as e:
        print(f"Error updating value in the table: {e}")


    finally:
        connection.close()
```

```
PS C:\Users\Akilesh K\OneDrive\Documents\python class\
hallenge\petpals'; & 'C:\Users\Akilesh K\AppData\Local
-2023.22.1\pythonFiles\lib\python\debugpy\adapter/../.
sign\techshop\tech.py'
enter product id:32
enter the new quantity:16
Updated value in the database.
```

```
mysql> select * from inventory;
+-------------+-----------+-----------------+-----------------+
| inventoryid | productid | quantityinstock | laststockupdate |
+-------------+-----------+-----------------+-----------------+
|        2679 |        23 |              25 | 2023-10-13      |
|        4235 |        32 |              16 | 2023-08-28      |
|        6746 |        73 |             100 | 2023-11-30      |
|        7452 |        44 |              10 | 2023-12-01      |
|        8363 |        65 |              30 | 2023-12-07      |
+-------------+-----------+-----------------+-----------------+
5 rows in set (0.00 sec)
```

6.

```
def get_order_summary():
    connection = create_connection()
    if connection:
        try:
            cursor = connection.cursor()
            productid=int(input("enter product number:"))
            select_query=("SELECT * FROM orderdetails where productid= %s")
```

```
        cursor.execute(select_query,(productid,))
        cus = cursor.fetchall()

        print("products:")

        print(cus)

    except Error as e:
        print(f"Error retrieving listings: {e}")
    finally:
        connection.close()
```

```
PS C:\Users\Akilesh K\OneDrive\Documents\python cl
hallenge\petpals'; & 'C:\Users\Akilesh K\AppData\L
-2023.22.1\pythonFiles\lib\python\debugpy\adapter/
sign\techshop\tech.py'
enter product number:32
products:
[(837, 331, 32, 10), (1026, 1021, 32, 4)]
```

7.

```
def update_customer():
    connection = create_connection()
    if connection:
        try:
            cursor = connection.cursor()
            customerid=int(input("enter customerid:"))
            new_phone=int(input("enter ne phone:"))
            update_query = ("UPDATE customers SET phone = %s WHERE customerid = %s")
            cursor.execute(update_query, (new_phone, customerid))
```

```
        connection.commit()
        print(f"Updated value in the database.")

    except Error as e:
        print(f"Error updating value in the table: {e}")

    finally:
        connection.close()
```

```
PS C:\Users\Akilesh K\OneDrive\Documents\python
hallenge\petpals'; & 'C:\Users\Akilesh K\AppDat
-2023.22.1\pythonFiles\lib\python\debugpy\adapte
sign\techshop\tech.py'
enter customerid:25
enter ne phone:52354636
Updated value in the database.
```

```
mysql> select * from customers;
+------------+-----------+----------+-------------------------+----------------+----------------------------+
| customerid | firstname | lastname | email                   | phone          | address                    |
+------------+-----------+----------+-------------------------+----------------+----------------------------+
|          2 | Benjamin  | May      | benjamin@icloud.net     | 1-573-762-5734 | Ap #774-4948 Hendrerit Av. |
|          5 | Aiko      | Barry    | aiko@outlook.com        | 1-344-918-8823 | 2780 Non, Street           |
|          7 | Burton    | Hicks    | burton2976@protonmail.ca | (889) 371-0537 | 924-9041 Nullam Rd.        |
|          9 | Lani      | Moon     | lani6109@protonmail.com | (225) 833-2432 | 743-1961 Sapien Road       |
|         10 | Jonah     | Boyle    | jonah2869@yahoo.org     | (555) 377-2867 | st george street,12        |
|         25 | aki       | k        | aki@gmail               | 52354636       | 4th cross                  |
+------------+-----------+----------+-------------------------+----------------+----------------------------+
6 rows in set (0.01 sec)
```

8.

```
def record_payment(order_id, payment_method, amount):
    connection = create_connection()
    if connection:
        try:
            cursor = connection.cursor()
            query = "INSERT INTO payments (order_id, payment_method, amount) VALUES (%s, %s, %s)"
            cursor.execute(query, (order_id, payment_method, amount))
            connection.commit()
```

```
        print("Payment recorded successfully.")

    except Error as e:
        connection.rollback()
        print(f"Error recording payment: {e}")

    finally:
        connection.close()
```

9.

```
def get_desc():
    connection = create_connection()
    if connection:
        try:
            cursor = connection.cursor()
            desc=(input("enter descriptions:"))
            select_query=("SELECT * FROM products where description= %s")
            cursor.execute(select_query,(desc,))
            cus = cursor.fetchall()

            print("products:")

            print(cus)

        except Error as e:
            print(f"Error retrieving listings: {e}")
        finally:
            connection.close()
```

```
PS C:\Users\Akilesh K\OneDrive\Documents\python cla
hallenge\petpals'; & 'C:\Users\Akilesh K\AppData\Lo
-2023.22.1\pythonFiles\lib\python\debugpy\adapter/.
sign\techshop\tech.py'
enter descriptions:pc
products:
[(32, 'imac', 'pc', 3000.0)]
```