

TABLE OF CONTENTS

Introduction	1
Source Code	5
Output	9
Changes in Tables	11
References	15

INTRODUCTION

This project, titled **Online Shopping Database Management System**, is designed to showcase the use of Python and MySQL for managing the backend operations of an e-commerce platform. It includes functionalities for handling user accounts, managing product inventories, and processing customer orders. The project demonstrates essential database operations like creating tables, inserting data, and performing CRUD (Create, Read, Update, Delete) operations.

Introduction:

In the modern digital era, online shopping has become an integral part of daily life. The purpose of this project is to simulate the backend of an online shopping platform. This system is developed to provide hands-on experience in designing and managing a database using Python and MySQL.

Key Features:

- **User Management:** Enables the creation, retrieval, and management of user accounts.
- **Product Management:** Handles product details such as names, descriptions, prices, stock levels, and associated brands.

- **Order Processing:** Simulates placing, viewing, and updating customer orders.

Objectives:

- To understand and implement PythonMySQL database connectivity.
- To design and manage relational databases effectively.
- To perform and demonstrate CRUD operations for practical use cases.

This project serves as a practical implementation of database concepts and offers insights into how real-world ecommerce systems manage their backend processes efficiently.

Also, the quantity of the product which has been ordered is reduced by 1 from the main table.

SOME IMPORTANT TERMS

- MySQL Database
- Interfacing MySQL with python
- MySQL connector
- Functions in python

SOURCE CODE

```
import mysql.connector
import pandas as pd

connection = mysql.connector.connect(host='localhost',user='root', password='1234567', database = 'main_products')
if connection.is_connected():
    print("Connected to MySQL database")

cursor = connection.cursor()
cursor.execute("CREATE DATABASE IF NOT EXISTS main_products")
print("Database 'main_products' created or already exists.")

def create_tables():

    cursor.execute('''CREATE TABLE IF NOT EXISTS Users (user_id INT AUTO_INCREMENT PRIMARY KEY,username
                    VARCHAR(50) NOT NULL,email VARCHAR(100) UNIQUE NOT NULL,password VARCHAR(100) NOT NULL);''')

    cursor.execute('''CREATE TABLE IF NOT EXISTS Products (product_id INT AUTO_INCREMENT PRIMARY KEY,product_name
                    VARCHAR(100) NOT NULL,description TEXT,price DECIMAL(10, 2) NOT NULL,stock INT NOT NULL,brand VARCHAR(50));''')

    cursor.execute('''CREATE TABLE IF NOT EXISTS Orders (order_id INT AUTO_INCREMENT PRIMARY KEY,user_id INT,product_id INT,
                    quantity INT NOT NULL,order_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,FOREIGN KEY (user_id) REFERENCES
                    Users(user_id),FOREIGN KEY (product_id) REFERENCES Products(product_id));''')

    connection.commit()
    print("Tables created successfully.")

create_tables()
```

```
#User Management
```

```
def add_user():
```

```
    username = input("Enter username: ")
```

```
    email = input("Enter email: ")
```

```
    password = input("Enter password: ")
```

```
    query = "INSERT INTO Users (username, email, password) VALUES (%s, %s, %s)"
```

```
    cursor.execute(query, (username, email, password))
```

```
    connection.commit()
```

```
    print("User added successfully.")
```

```
def view_users():
```

```
    cursor.execute("SELECT * FROM Users")
```

```
    data = cursor.fetchall()
```

```
    df = pd.DataFrame(data, columns=['user_id', 'username', 'email', 'password', 'created_at'])
```

```
    print(df)
```

```
def update_user():
```

```
    user_id = input("Enter User ID to update: ")
```

```
    new_email = input("Enter new email: ")
```

```
    query = "UPDATE Users SET email = %s WHERE user_id = %s"
```

```
    cursor.execute(query, (new_email, user_id))
```

```
    connection.commit()
```

```
    print("User updated successfully.")
```

```
def delete_user():
```

```
    user_id = input("Enter User ID to delete: ")
```

```
    query = "DELETE FROM Users WHERE user_id = %s"
```

```
    cursor.execute(query, (user_id,))
```

```
    connection.commit()
```

```
    print("User deleted successfully.")
```

```

# Product Management
def add_product():
    product_name = input("Enter product name: ")
    description = input("Enter product description: ")
    price = float(input("Enter product price: "))
    stock = int(input("Enter stock quantity: "))
    query = "INSERT INTO Products (product_name, description, price, stock) VALUES (%s, %s, %s, %s)"
    cursor.execute(query, (product_name, description, price, stock, brand))
    connection.commit()
    print("Product added successfully.")

def view_products():
    cursor.execute("SELECT * FROM Products")
    data = cursor.fetchall()
    df = pd.DataFrame(data, columns=['product_id', 'product_name', 'description', 'price', 'stock', 'brand', 'created_at'])
    print(df)

def update_product():
    product_id = input("Enter Product ID to update: ")
    new_price = float(input("Enter new price: "))
    query = "UPDATE Products SET price = %s WHERE product_id = %s"
    cursor.execute(query, (new_price, product_id))
    connection.commit()
    print("Product updated successfully.")

def delete_product():
    product_id = input("Enter Product ID to delete: ")
    query = "DELETE FROM Products WHERE product_id = %s"
    cursor.execute(query, (product_id,))
    connection.commit()
    print("Product deleted successfully.")

# Order Management
def place_order():
    user_id = input("Enter User ID: ")
    product_id = input("Enter Product ID: ")
    quantity = int(input("Enter quantity: "))
    query = "INSERT INTO Orders (user_id, product_id, quantity) VALUES (%s, %s, %s)"
    cursor.execute(query, (user_id, product_id, quantity))
    connection.commit()
    print("Order placed successfully.")

def view_orders():
    cursor.execute("SELECT * FROM Orders")
    data = cursor.fetchall()
    df = pd.DataFrame(data, columns=['order_id', 'user_id', 'product_id', 'quantity', 'order_date'])
    print(df)

def update_order():
    order_id = input("Enter Order ID to update: ")
    new_quantity = int(input("Enter new quantity: "))
    query = "UPDATE Orders SET quantity = %s WHERE order_id = %s"
    cursor.execute(query, (new_quantity, order_id))
    connection.commit()
    print("Order updated successfully.")

def delete_order():
    order_id = input("Enter Order ID to delete: ")
    query = "DELETE FROM Orders WHERE order_id = %s"
    cursor.execute(query, (order_id,))
    connection.commit()
    print("Order deleted successfully.")

```

```
# Database Operations
def display_tables():
    cursor.execute("SHOW TABLES")
    tables = cursor.fetchall()
    print("Available tables:")
    for table in tables:
        print(table[0])
def reset_database():
    confirmation = input("Are you sure you want to reset the database? This will delete all data! (yes/no): ")
    if confirmation.lower() == 'yes':
        cursor.execute("DELETE FROM Users")
        cursor.execute("DELETE FROM Products")
        cursor.execute("DELETE FROM Orders")
        connection.commit()
        print("Database reset successfully.")
```

```

# Main Menu
def main_menu():
    while True:
        print("\n=== Online Shopping Database Management System ===")
        print("1. User Management")
        print("2. Product Management")
        print("3. Order Management")
        print("4. Database Operations")
        print("5. Exit")
        choice = input("Enter your choice (1-5): ")
        if choice == '1':
            print("\n--- Us New User")
            print("er Management ---")
            print("1. Add a2. View All Users")
            print("3. Update User Details")
            print("4. Delete a User")
            user_choice = input("Enter your choice (1-4): ")
            if user_choice == '1':
                add_user()
            elif user_choice == '2':
                view_users()
            elif user_choice == '3':
                update_user()
            elif user_choice == '4':
                delete_user()
            else:
                print("Invalid choice.")
        elif choice == '2':
            print("\n--- Product Management ---")
            print("1. Add a New Product")
            print("2. View All Products")
            print("3. Update Product Details")
            print("4. Delete a Product")
            product_choice = input("Enter your choice (1-4): ")
            if product_choice == '1':
                add_product()
            elif product_choice == '2':
                view_products()
            elif product_choice == '3':
                update_product()
            elif product_choice == '4':
                delete_product()

```

```
    else:
        print("Invalid choice.")
elif choice == '3':
    print("\n--- Order Management ---")
    print("1. Place a New Order")
    print("2. View All Orders")
    print("3. Update an Order")
    print("4. Delete an Order")
    order_choice = input("Enter your choice (1-4): ")
    if order_choice == '1':
        place_order()
    elif order_choice == '2':
        view_orders()
    elif order_choice == '3':
        update_order()
    elif order_choice == '4':
        delete_order()
    else:
        print("Invalid choice.")
```

```
elif choice == '4':
    print("\n--- Database Operations ---")
    print("1. Display Database Tables")
    print("2. Reset Database")
    db_choice = input("Enter your choice (1-2): ")
    if db_choice == '1':
        display_tables()
    elif db_choice == '2':
        reset_database()
    else:
        print("Invalid choice.")
elif choice == '5':
    print("Exiting the program. Goodbye!")
    break
else:
    print("Invalid choice. Please try again.")

main_menu()

cursor.close()
connection.close()
```

OUTPUT

ADDING USERS

```
Connected to MySQL database
Database 'main_products' created or already exists.
Tables created successfully.
```

```
=== Online Shopping Database Management System ===
```

```
1. User Management
2. Product Management
3. Order Management
4. Database Operations
5. Exit
Enter your choice (1-5): 1
```

```
--- User Management ---
```

```
1. Add a new user
2. View All Users
3. Update User Details
4. Delete a User
Enter your choice (1-4): 1
Enter username: AdityaSharma
Enter email: adityasharma@example.com
Enter password: securepassword
User added successfully.
```

```
=== Online Shopping Database Management System ===
```

```
1. User Management
2. Product Management
3. Order Management
4. Database Operations
5. Exit
Enter your choice (1-5): 1
```

```
--- User Management ---
```

```
1. Add a new user
2. View All Users
3. Update User Details
4. Delete a User
Enter your choice (1-4): 1
Enter username: Adi
Enter email: aditya@csproject.com
Enter password: anotherpassword
User added successfully.
```

ADDING PRODUCTS

Connected to MySQL database
Database 'main_products' created or already exists.
Tables created successfully.

=== Online Shopping Database Management System ===

1. User Management
2. Product Management
3. Order Management
4. Database Operations
5. Exit

Enter your choice (1-5): 2

--- Product Management ---

1. Add a New Product
2. View All Products
3. Update Product Details
4. Delete a Product

Enter your choice (1-4): 1

Enter product name: Laptop

Enter product description: High-performance laptop

Enter product price: 75000.0

Enter stock quantity: 8

Product added successfully.

=== Online Shopping Database Management System ===

1. User Management
2. Product Management
3. Order Management
4. Database Operations
5. Exit

Enter your choice (1-5): 2

--- Product Management ---

1. Add a New Product
2. View All Products
3. Update Product Details
4. Delete a Product

Enter your choice (1-4): 1

Enter product name: Smartphone

Enter product description: Latest model smartphone

Enter product price: 50000.0

Enter stock quantity: 15

Product added successfully.

PLACING ORDER

=== Online Shopping Database Management System ===

1. User Management
2. Product Management
3. Order Management
4. Database Operations
5. Exit

Enter your choice (1-5): 3

--- Order Management ---

1. Place a New Order
2. View All Orders
3. Update an Order
4. Delete an Order

Enter your choice (1-4): 1

Enter User ID: 1

Enter Product ID: 1

Enter quantity: 2

Order placed successfully.

CHANGES IN TABLES

USER TABLE

```
mysql> select * from users;
```

user_id	username	email	password	created_at
1	Adi20	adityasharma@csproject.com	cs2025	2024-12-06 09:55:04
3	AdityaSharma	adityasharma@example.com	securepassword	2024-12-10 09:47:02
4	Adi	aditya@csproject.com	anotherpassword	2024-12-10 09:47:27

3 rows in set (0.01 sec)

PRODUCT TABLE

```
mysql> select * from products;
```

product_id	product_name	description	price	stock	brand	created_at
1	Macbook	Apple M4 Macbook	170000.00	2	Apple	2024-12-06 09:56:22
2	Laptop	High-performance laptop	75000.00	8	NULL	2024-12-10 09:58:56
3	Smartphone	Latest model smartphone	50000.00	15	NULL	2024-12-10 09:59:32

3 rows in set (0.01 sec)

REFERENCES

Class 11 Book (Sumitra Arora python)

Class 12 Book (Sumitra Arora Python)
