Coding Challenge: Order Management System

Project Overview

The Order Management System is a console-based application built using C# and SQL Server. It demonstrates CRUD operations, database interaction and exception handling. Users can manage products and orders through a menu.

Technologies Used

- C# (.NET Framework)
- SQL Server (LocalDB)
- Visual Studio 2022
- ADO.NET for database interaction

Directory Structure

- model – Classes like Product, Electronics, Clothing, User.
- Interface and implementation (IOrderManagement, OrderProcessor)
- exception – UserNotFoundException, OrderNotFoundException.
- util – Utility classes for DB connection (DBConnectUtil, DBPropertyUtil)
- main – Main.cs

Creating the database:

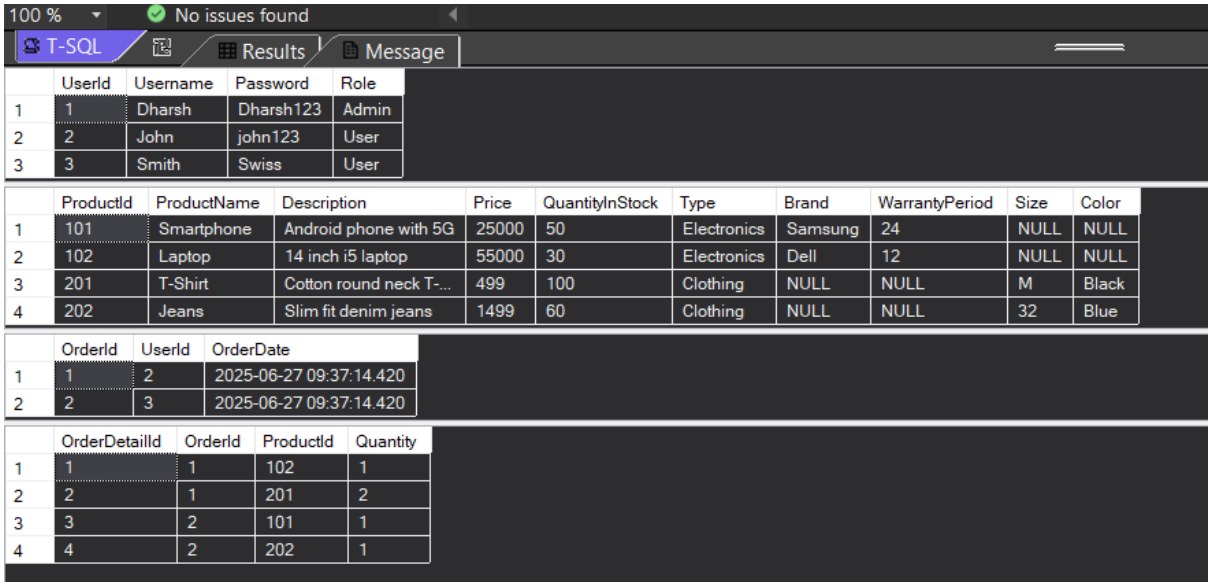
```
CREATE DATABASE OrderManagementSystem

USE OrderManagementSystem

INSERT INTO Users (UserId, Username, Password, Role)
VALUES
(1, 'Dharsh', 'Dharsh123', 'Admin'),
(2, 'John', 'john123', 'User'),
(3, 'Jane', 'jane456', 'User');

INSERT INTO Products
(ProductId, ProductName, Description, Price, QuantityInStock, Type, Brand, WarrantyPeriod, Size, Color)
VALUES
(101, 'Smartphone', 'Android phone with 5G', 25000.00, 50, 'Electronics', 'Samsung', 24, NULL, NULL),
(102, 'Laptop', '14 inch i5 laptop', 55000.00, 30, 'Electronics', 'Dell', 12, NULL, NULL);

INSERT INTO Products
(ProductId, ProductName, Description, Price, QuantityInStock, Type, Brand, WarrantyPeriod, Size, Color)
VALUES
(201, 'T-Shirt', 'Cotton round neck T-shirt', 499.00, 100, 'Clothing', NULL, NULL, 'M', 'Black'),
```

(202, 'Jeans', 'Slim fit denim jeans', 1499.00, 60, 'Clothing', NULL, NULL, '32', 'Blue');

INSERT INTO OrderDetails (OrderId, ProductId, Quantity)
VALUES
(1, 102, 1),
(1, 201, 2);

INSERT INTO OrderDetails (OrderId, ProductId, Quantity)
VALUES
(2, 101, 1),
(2, 202, 1);

SELECT * FROM Users;
SELECT * FROM Products;
SELECT * FROM Orders;
SELECT * FROM OrderDetails;



100 % No issues found

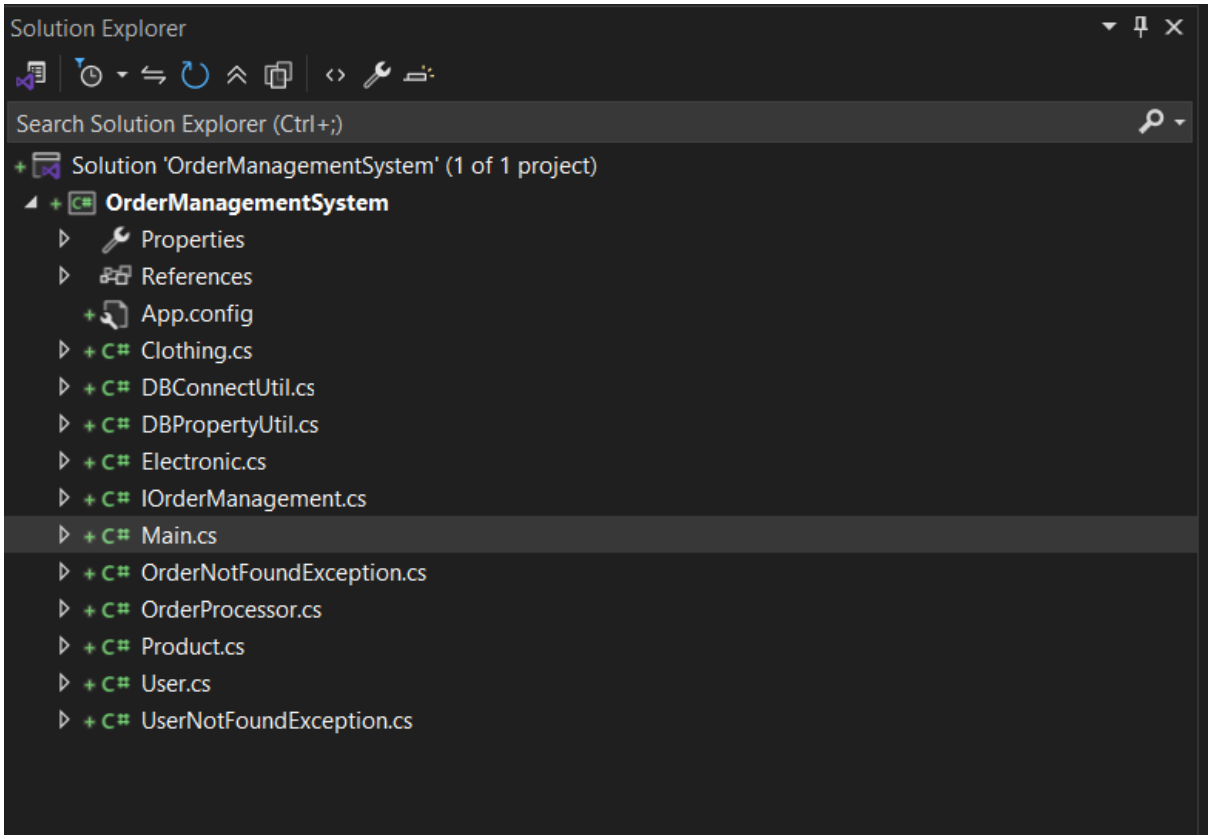
T-SQL Results Message

	UserId	Username	Password	Role
1	1	Dharsh	Dharsh123	Admin
2	2	John	john123	User
3	3	Smith	Swiss	User

	ProductId	ProductName	Description	Price	QuantityInStock	Type	Brand	WarrantyPeriod	Size	Color
1	101	Smartphone	Android phone with 5G	25000	50	Electronics	Samsung	24	NULL	NULL
2	102	Laptop	14 inch i5 laptop	55000	30	Electronics	Dell	12	NULL	NULL
3	201	T-Shirt	Cotton round neck T-...	499	100	Clothing	NULL	NULL	M	Black
4	202	Jeans	Slim fit denim jeans	1499	60	Clothing	NULL	NULL	32	Blue

	OrderId	UserId	OrderDate
1	1	2	2025-06-27 09:37:14.420
2	2	3	2025-06-27 09:37:14.420

	OrderDetailId	OrderId	ProductId	Quantity
1	1	1	102	1
2	2	1	201	2
3	3	2	101	1
4	4	2	202	1



Product.cs

```
public class Product
{
    public int ProductId { get; set; }
    public string ProductName { get; set; }
    public string Description { get; set; }
    public double Price { get; set; }
    public int QuantityInStock { get; set; }
    public string Type { get; set; }

    public Product() {}

    public Product(int id, string name, string desc, double price, int qty, string type)
    {
        ProductId = id;
        ProductName = name;
        Description = desc;
        Price = price;
        QuantityInStock = qty;
        Type = type;
    }
}
```

Electronics.cs

```
public class Electronics : Product
{
    public string Brand { get; set; }
    public int WarrantyPeriod { get; set; }

    public Electronics(int id, string name, string desc, double price, int qty, string brand, int warranty)
        : base(id, name, desc, price, qty, "Electronics")
    {
        Brand = brand;
        WarrantyPeriod = warranty;
    }
}
```

Clothing.cs

```
public class Clothing : Product
{
    public string Size { get; set; }
    public string Color { get; set; }

    public Clothing(int id, string name, string desc, double price, int qty, string size, string color)
        : base(id, name, desc, price, qty, "Clothing")
    {
        Size = size;
        Color = color;
    }
}
```

User.cs

```
public class User
{
    public int UserId { get; set; }
    public string Username { get; set; }
    public string Password { get; set; }
    public string Role { get; set; }

    public User(int id, string uname, string pwd, string role)
    {
        UserId = id;
        Username = uname;
        Password = pwd;
        Role = role;
    }
}
```

IOrderManagementRepository.cs

```
using System.Collections.Generic;

public interface IOrderManagementRepository
{
   void CreateUser(User user);
   void CreateProduct(User user, Product product);
   void CreateOrder(User user, List<Product> products);
   void CancelOrder(int userId, int orderId);
   List<Product> GetAllProducts();
   List<Product> GetOrderByUser(User user);
}
```

OrderProcessor.cs

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;

public class OrderProcessor : IOrderManagementRepository
{
   private const string ConnectionName = "OrderDB";

   public void CreateUser(User user)
   {
      using (SqlConnection conn = DBConnUtil.GetConnection(ConnectionName))
      {
         conn.Open();
         string query = "INSERT INTO Users (UserId, Username, Password, Role) VALUES (@id,
@username, @password, @role)";
         SqlCommand cmd = new SqlCommand(query, conn);
         cmd.Parameters.AddWithValue("@id", user.UserId);
         cmd.Parameters.AddWithValue("@username", user.Username);
         cmd.Parameters.AddWithValue("@password", user.Password);
         cmd.Parameters.AddWithValue("@role", user.Role);
         cmd.ExecuteNonQuery();

         Console.WriteLine("User created successfully.");
      }
   }

   public void CreateProduct(User user, Product product)
   {
      if (!user.Role.Equals("Admin", StringComparison.OrdinalIgnoreCase))
      {
         Console.WriteLine("Only admin users can create products.");
         return;
```

```
    }

    using (SqlConnection conn = DBConnUtil.GetConnection(ConnectionName))
    {
      conn.Open();
      string query = @"INSERT INTO Products
            (ProductId, ProductName, Description, Price, QuantityInStock, Type, Brand,
WarrantyPeriod, Size, Color)
            VALUES
            (@id, @name, @desc, @price, @qty, @type, @brand, @warranty, @size, @color)";

      SqlCommand cmd = new SqlCommand(query, conn);
      cmd.Parameters.AddWithValue("@id", product.ProductId);
      cmd.Parameters.AddWithValue("@name", product.ProductName);
      cmd.Parameters.AddWithValue("@desc", product.Description);
      cmd.Parameters.AddWithValue("@price", product.Price);
      cmd.Parameters.AddWithValue("@qty", product.QuantityInStock);
      cmd.Parameters.AddWithValue("@type", product.Type);

      cmd.Parameters.AddWithValue("@brand", GetProperty(product, "Brand") ?? DBNull.Value);
      cmd.Parameters.AddWithValue("@warranty", GetProperty(product, "WarrantyPeriod") ??
DBNull.Value);
      cmd.Parameters.AddWithValue("@size", GetProperty(product, "Size") ?? DBNull.Value);
      cmd.Parameters.AddWithValue("@color", GetProperty(product, "Color") ?? DBNull.Value);

      cmd.ExecuteNonQuery();
      Console.WriteLine("Product created successfully.");
    }
  }

  public void CreateOrder(User user, List<Product> products)
  {
    using (SqlConnection conn = DBConnUtil.GetConnection(ConnectionName))
    {
      conn.Open();
      SqlTransaction transaction = conn.BeginTransaction();

      try
      {
        SqlCommand checkUser = new SqlCommand("SELECT COUNT(*) FROM Users WHERE UserId
= @uid", conn, transaction);
        checkUser.Parameters.AddWithValue("@uid", user.UserId);
        int count = (int)checkUser.ExecuteScalar();

        if (count == 0)
        {
          SqlCommand insertUser = new SqlCommand("INSERT INTO Users (UserId, Username,
Password, Role) VALUES (@id, @username, @password, @role)", conn, transaction);
          insertUser.Parameters.AddWithValue("@id", user.UserId);
          insertUser.Parameters.AddWithValue("@username", user.Username);
          insertUser.Parameters.AddWithValue("@password", user.Password);
```

```
                insertUser.Parameters.AddWithValue("@role", user.Role);
                insertUser.ExecuteNonQuery();
            }

            SqlCommand insertOrder = new SqlCommand("INSERT INTO Orders (UserId) OUTPUT
INSERTED.OrderId VALUES (@uid)", conn, transaction);
            insertOrder.Parameters.AddWithValue("@uid", user.UserId);
            int orderId = (int)insertOrder.ExecuteScalar();

            foreach (Product product in products)
            {
                SqlCommand insertDetail = new SqlCommand("INSERT INTO OrderDetails (OrderId,
ProductId, Quantity) VALUES (@oid, @pid, 1)", conn, transaction);
                insertDetail.Parameters.AddWithValue("@oid", orderId);
                insertDetail.Parameters.AddWithValue("@pid", product.ProductId);
                insertDetail.ExecuteNonQuery();
            }

            transaction.Commit();
            Console.WriteLine("Order placed successfully. Order ID: " + orderId);
        }
        catch (Exception ex)
        {
            transaction.Rollback();
            Console.WriteLine("Failed to place order: " + ex.Message);
        }
    }
  }

  public void CancelOrder(int userId, int orderId)
  {
    using (SqlConnection conn = DBConnUtil.GetConnection(ConnectionName))
    {
        conn.Open();
        SqlTransaction transaction = conn.BeginTransaction();

        try
        {
            SqlCommand checkUser = new SqlCommand("SELECT COUNT(*) FROM Users WHERE UserId
= @uid", conn, transaction);
            checkUser.Parameters.AddWithValue("@uid", userId);
            if ((int)checkUser.ExecuteScalar() == 0) throw new UserNotFoundException("User not
found.");

            SqlCommand checkOrder = new SqlCommand("SELECT COUNT(*) FROM Orders WHERE
OrderId = @oid AND UserId = @uid", conn, transaction);
            checkOrder.Parameters.AddWithValue("@oid", orderId);
            checkOrder.Parameters.AddWithValue("@uid", userId);
            if ((int)checkOrder.ExecuteScalar() == 0) throw new OrderNotFoundException("Order not
found.");
```

```
            SqlCommand deleteDetails = new SqlCommand("DELETE FROM OrderDetails WHERE
OrderId = @oid", conn, transaction);
            deleteDetails.Parameters.AddWithValue("@oid", orderId);
            deleteDetails.ExecuteNonQuery();

            SqlCommand deleteOrder = new SqlCommand("DELETE FROM Orders WHERE OrderId =
@oid", conn, transaction);
            deleteOrder.Parameters.AddWithValue("@oid", orderId);
            deleteOrder.ExecuteNonQuery();

            transaction.Commit();
            Console.WriteLine("Order cancelled successfully.");
        }
        catch (Exception ex)
        {
            transaction.Rollback();
            Console.WriteLine("Cancellation failed: " + ex.Message);
        }
    }
  }

  public List<Product> GetAllProducts()
  {
    return FetchProducts("SELECT * FROM Products", null);
  }

  public List<Product> GetOrderByUser(User user)
  {
    string query = @"SELECT P.* FROM Orders O
                JOIN OrderDetails OD ON O.OrderId = OD.OrderId
                JOIN Products P ON OD.ProductId = P.ProductId
                WHERE O.UserId = @uid";

    SqlParameter[] parameters = { new SqlParameter("@uid", user.UserId) };
    return FetchProducts(query, parameters);
  }

  private List<Product> FetchProducts(string query, SqlParameter[] parameters)
  {
    List<Product> products = new List<Product>();
    using (SqlConnection conn = DBConnUtil.GetConnection(ConnectionName))
    {
      conn.Open();
      SqlCommand cmd = new SqlCommand(query, conn);
      if (parameters != null) cmd.Parameters.AddRange(parameters);

      SqlDataReader reader = cmd.ExecuteReader();
      while (reader.Read())
      {
        products.Add(MapReaderToProduct(reader));
      }
```

```
        }
        return products;
    }

    private Product MapReaderToProduct(SqlDataReader reader)
    {
        string type = reader["Type"].ToString().ToLower();
        if (type == "electronics")
        {
            return new Electronics(
                Convert.ToInt32(reader["ProductId"]),
                reader["ProductName"].ToString(),
                reader["Description"].ToString(),
                Convert.ToDouble(reader["Price"]),
                Convert.ToInt32(reader["QuantityInStock"]),
                reader["Brand"].ToString(),
                Convert.ToInt32(reader["WarrantyPeriod"])
            );
        }
        else if (type == "clothing")
        {
            return new Clothing(
                Convert.ToInt32(reader["ProductId"]),
                reader["ProductName"].ToString(),
                reader["Description"].ToString(),
                Convert.ToDouble(reader["Price"]),
                Convert.ToInt32(reader["QuantityInStock"]),
                reader["Size"].ToString(),
                reader["Color"].ToString()
            );
        }
        else
        {
            return new Product(
                Convert.ToInt32(reader["ProductId"]),
                reader["ProductName"].ToString(),
                reader["Description"].ToString(),
                Convert.ToDouble(reader["Price"]),
                Convert.ToInt32(reader["QuantityInStock"]),
                type
            );
        }
    }

    private object GetProperty(object obj, string propertyName)
    {
        var prop = obj.GetType().GetProperty(propertyName);
        return prop?.GetValue(obj);
    }
}
```

EXCEPTIONS

UserNotFoundException.cs

```
using System;

public class UserNotFoundException : Exception
{
   public UserNotFoundException(string message) : base(message) {}
}
```

OrderNotFoundException.cs

```
using System;

public class OrderNotFoundException : Exception
{
   public OrderNotFoundException(string message) : base(message) {}
}
```

UTIL CLASSES

DBPropertyUtil.cs

```
public static class DBPropertyUtil
{
   public static string GetConnectionString(string name)
   {
      return "Data Source=(localdb)\\MSSQLLocalDB;Initial
Catalog=OrderManagementSystem;Integrated Security=True";
   }
}
```

DBConnectUtil.cs

```
using System.Data.SqlClient;

public static class DBConnUtil
{
   public static SqlConnection GetConnection(string name)
   {
      string connString = DBPropertyUtil.GetConnectionString(name);
      return new SqlConnection(connString);
   }
}
```

MAIN MODULE

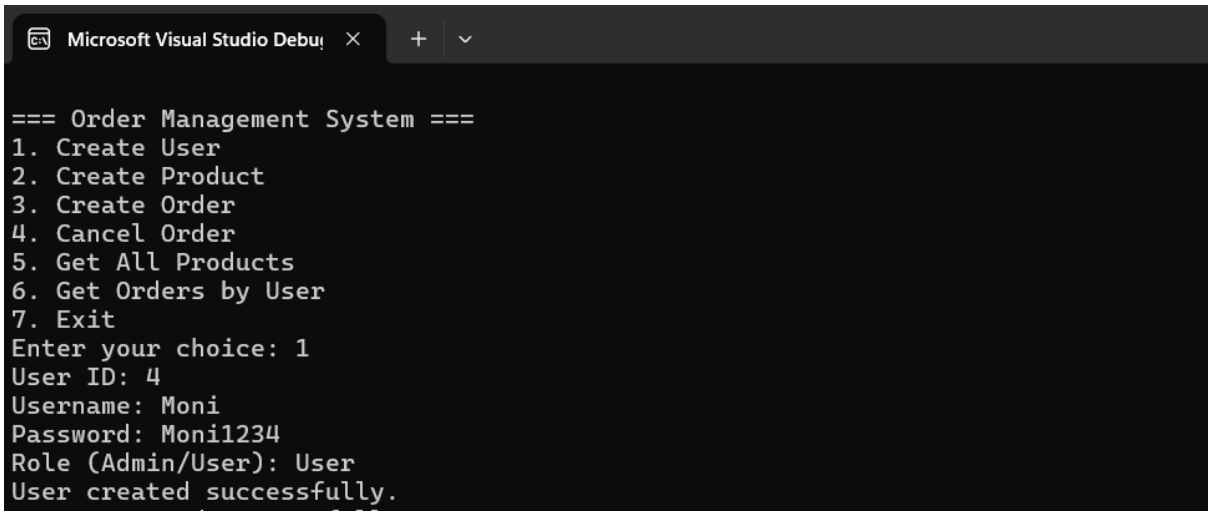
```
using System;
using System.Collections.Generic;

public class MainModule
{
   public static void Main(string[] args)
   {
      IOrderManagementRepository orderRepo = new OrderProcessor();

      while (true)
      {
         Console.WriteLine("\n=== Order Management System ===");
         Console.WriteLine("1. Create User");
         Console.WriteLine("2. Create Product");
         Console.WriteLine("3. Create Order");
         Console.WriteLine("4. Cancel Order");
         Console.WriteLine("5. Get All Products");
         Console.WriteLine("6. Get Orders by User");
         Console.WriteLine("7. Exit");
         Console.Write("Enter your choice: ");

         int choice = Convert.ToInt32(Console.ReadLine());

         try
         {
            switch (choice)
            {
               case 1:
                  Console.Write("User ID: ");
                  int userId = Convert.ToInt32(Console.ReadLine());
                  Console.Write("Username: ");
                  string username = Console.ReadLine();
                  Console.Write("Password: ");
                  string password = Console.ReadLine();
                  Console.Write("Role (Admin/User): ");
                  string role = Console.ReadLine();

                  User newUser = new User(userId, username, password, role);
                  orderRepo.CreateUser(newUser);
                  Console.WriteLine(" User created successfully.");
                  break;

               case 2:
                  Console.Write("User ID (Admin): ");
                  int adminId = Convert.ToInt32(Console.ReadLine());
                  Console.Write("Username: ");
                  string adminName = Console.ReadLine();
                  Console.Write("Password: ");
                  string adminPwd = Console.ReadLine();
```

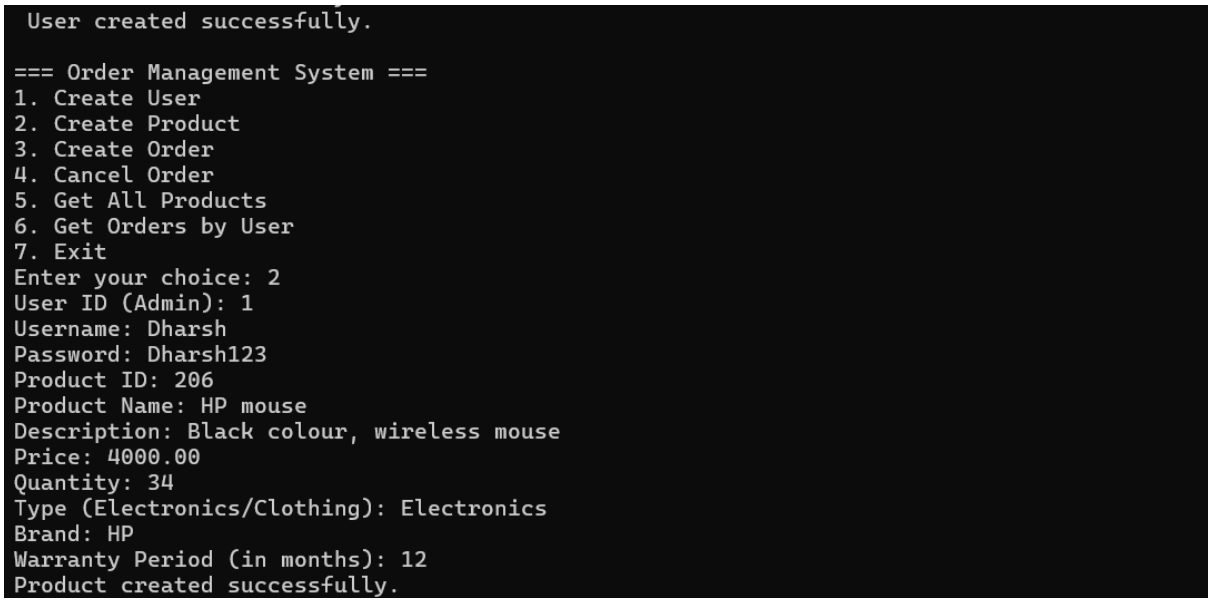
```
    User admin = new User(adminId, adminName, adminPwd, "Admin");

    Console.Write("Product ID: ");
    int pid = Convert.ToInt32(Console.ReadLine());
    Console.Write("Product Name: ");
    string pname = Console.ReadLine();
    Console.Write("Description: ");
    string desc = Console.ReadLine();
    Console.Write("Price: ");
    double price = Convert.ToDouble(Console.ReadLine());
    Console.Write("Quantity: ");
    int qty = Convert.ToInt32(Console.ReadLine());
    Console.Write("Type (Electronics/Clothing): ");
    string type = Console.ReadLine();

    Product product = null;

    if (type.ToLower() == "electronics")
    {
        Console.Write("Brand: ");
        string brand = Console.ReadLine();
        Console.Write("Warranty Period (in months): ");
        int warranty = Convert.ToInt32(Console.ReadLine());
        product = new Electronics(pid, pname, desc, price, qty, brand, warranty);
    }
    else if (type.ToLower() == "clothing")
    {
        Console.Write("Size: ");
        string size = Console.ReadLine();
        Console.Write("Color: ");
        string color = Console.ReadLine();
        product = new Clothing(pid, pname, desc, price, qty, size, color);
    }
    else
    {
        product = new Product(pid, pname, desc, price, qty, type);
    }

    orderRepo.CreateProduct(admin, product);

    break;

case 3:
    Console.Write("User ID: ");
    int uid = Convert.ToInt32(Console.ReadLine());
    Console.Write("Username: ");
    string uname = Console.ReadLine();
    Console.Write("Password: ");
    string pwd = Console.ReadLine();
```

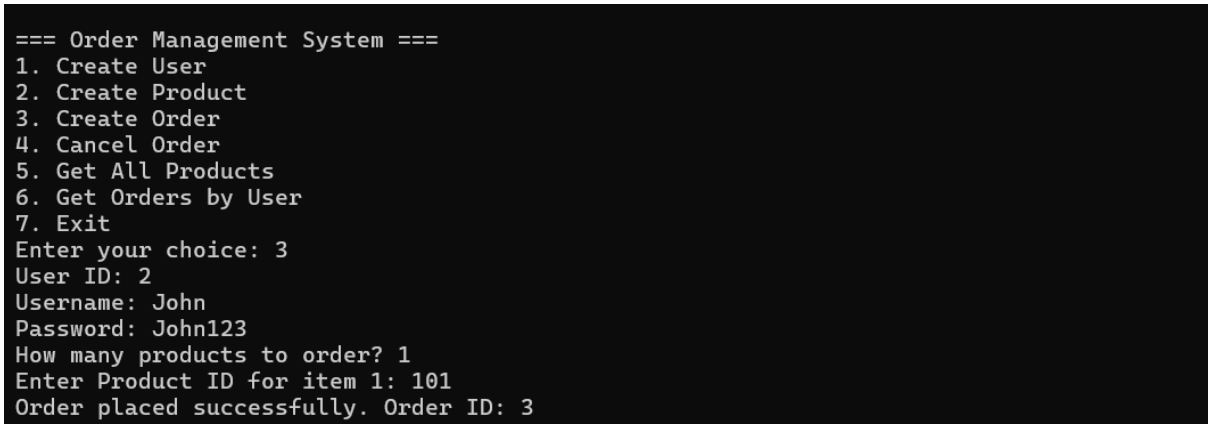
```
                User orderUser = new User(uid, uname, pwd, "User");

                List<Product> productList = new List<Product>();

                Console.Write("How many products to order? ");
                int count = Convert.ToInt32(Console.ReadLine());

                for (int i = 0; i < count; i++)
                {
                    Console.Write("Enter Product ID for item " + (i + 1) + ": ");
                    int prId = Convert.ToInt32(Console.ReadLine());


                    productList.Add(new Product { ProductId = prId });
                }

                orderRepo.CreateOrder(orderUser, productList);
                Console.WriteLine("Order created successfully.");
                break;

            case 4:
                Console.Write("Enter User ID: ");
                int cancelUserId = Convert.ToInt32(Console.ReadLine());
                Console.Write("Enter Order ID to cancel: ");
                int cancelOrderId = Convert.ToInt32(Console.ReadLine());

                orderRepo.CancelOrder(cancelUserId, cancelOrderId);
                Console.WriteLine("Order cancelled successfully.");
                break;

            case 5:
                List<Product> allProducts = orderRepo.GetAllProducts();
                Console.WriteLine("\n All Products:");
                foreach (Product prod in allProducts)
                {
                    Console.WriteLine($"ID: {prod.ProductId}, Name: {prod.ProductName}, Price:
{prod.Price}, Type: {prod.Type}");
                }
                break;

            case 6:
                Console.Write("Enter User ID to fetch orders: ");
                int findUserId = Convert.ToInt32(Console.ReadLine());
                User u = new User(findUserId, "", "", "User");

                List<Product> orderedProducts = orderRepo.GetOrderByUser(u);
                Console.WriteLine("\nOrders by User ID " + findUserId + ":");
                foreach (Product p in orderedProducts)
                {
                    Console.WriteLine($"Product ID: {p.ProductId}, Name: {p.ProductName}, Type:
{p.Type}");
```

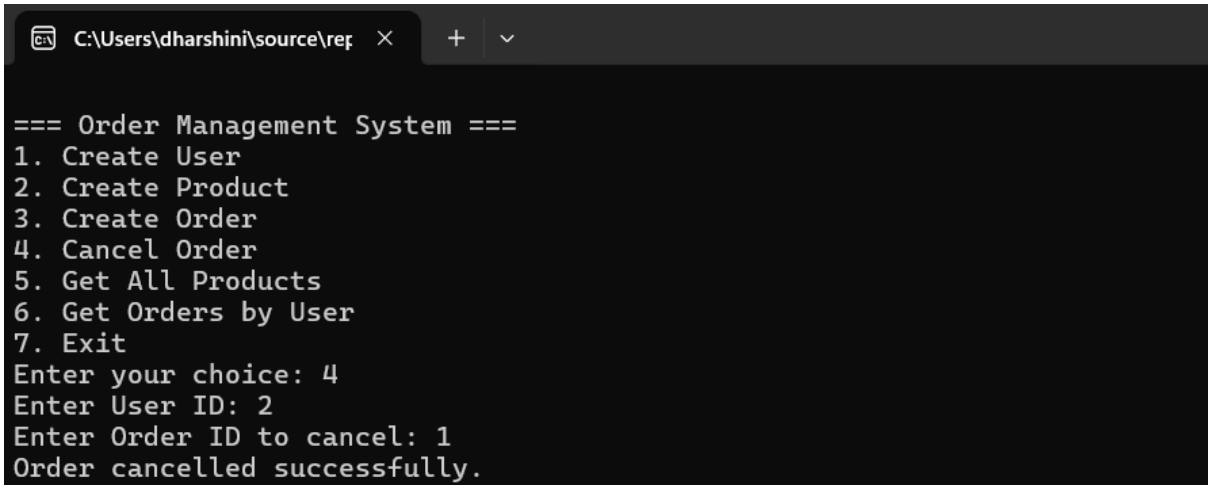
```
                }
                break;

            case 7:
                Console.WriteLine("Exiting Order Management System.");
                return;

            default:
                Console.WriteLine("Invalid choice. Try again.");
                break;
        }
      }
      catch (Exception ex)
      {
        Console.WriteLine(" Error: " + ex.Message);
      }
    }
  }
}
```

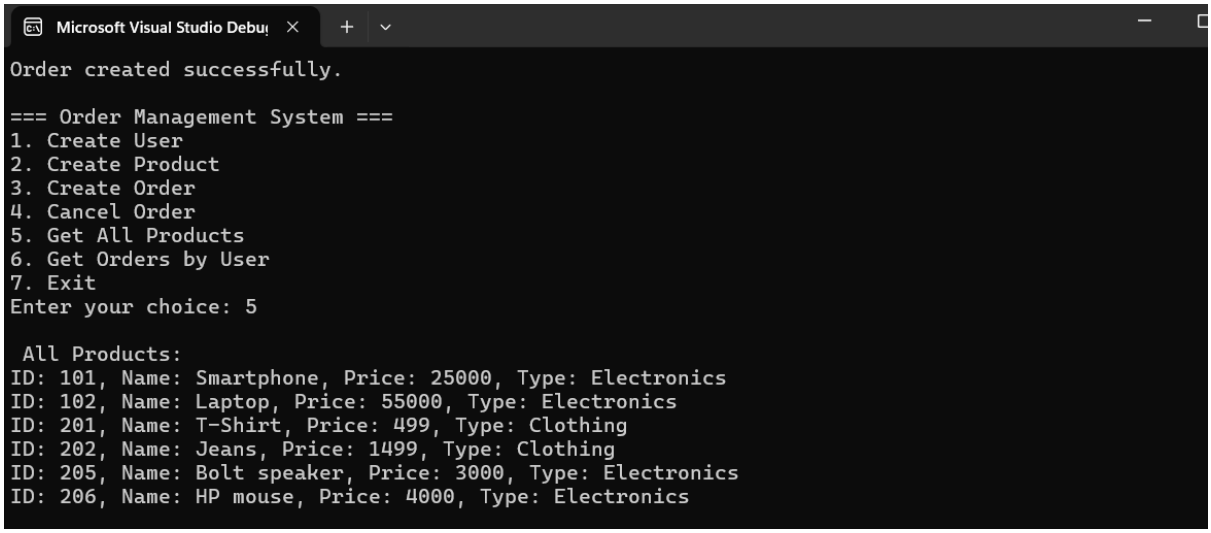
Output:

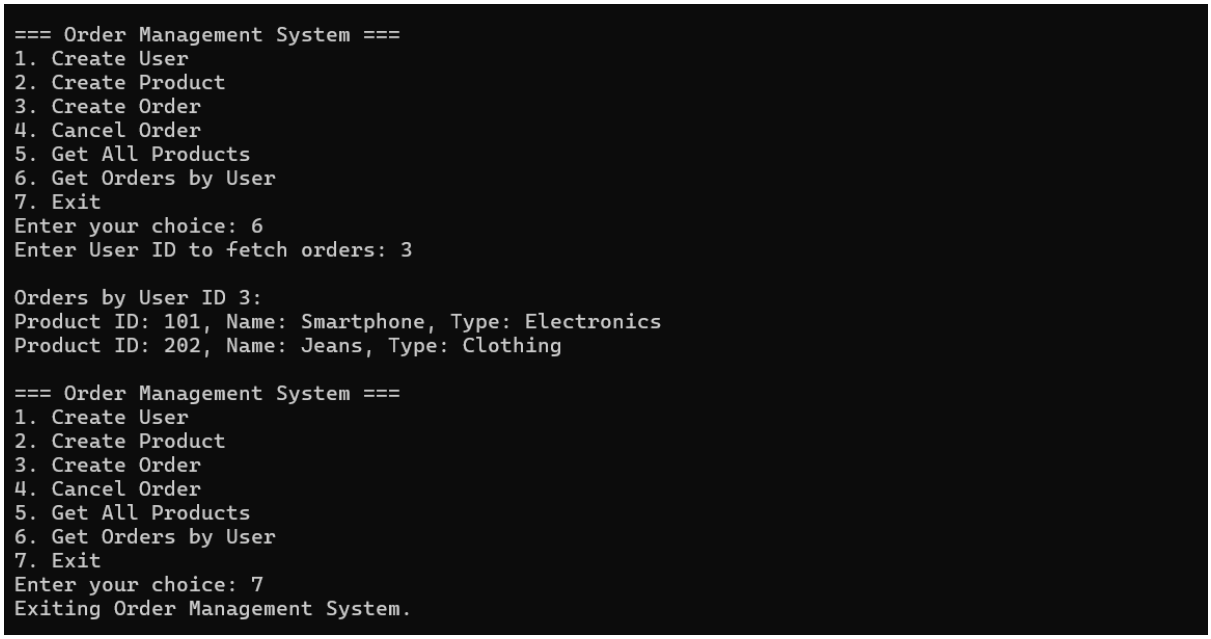

Microsoft Visual Studio Debu
=== Order Management System ===
1. Create User
2. Create Product
3. Create Order
4. Cancel Order
5. Get All Products
6. Get Orders by User
7. Exit
Enter your choice: 1
User ID: 4
Username: Moni
Password: Moni1234
Role (Admin/User): User
User created successfully.


User created successfully.
=== Order Management System ===
1. Create User
2. Create Product
3. Create Order
4. Cancel Order
5. Get All Products
6. Get Orders by User
7. Exit
Enter your choice: 2
User ID (Admin): 1
Username: Dharsh
Password: Dharsh123
Product ID: 206
Product Name: HP mouse
Description: Black colour, wireless mouse
Price: 4000.00
Quantity: 34
Type (Electronics/Clothing): Electronics
Brand: HP
Warranty Period (in months): 12
Product created successfully.


=== Order Management System ===
1. Create User
2. Create Product
3. Create Order
4. Cancel Order
5. Get All Products
6. Get Orders by User
7. Exit
Enter your choice: 3
User ID: 2
Username: John
Password: John123
How many products to order? 1
Enter Product ID for item 1: 101
Order placed successfully. Order ID: 3


C:\Users\dharshini\source\rep
=== Order Management System ===
1. Create User
2. Create Product
3. Create Order
4. Cancel Order
5. Get All Products
6. Get Orders by User
7. Exit
Enter your choice: 4
Enter User ID: 2
Enter Order ID to cancel: 1
Order cancelled successfully.


Microsoft Visual Studio Debu
Order created successfully.
=== Order Management System ===
1. Create User
2. Create Product
3. Create Order
4. Cancel Order
5. Get All Products
6. Get Orders by User
7. Exit
Enter your choice: 5
All Products:
ID: 101, Name: Smartphone, Price: 25000, Type: Electronics
ID: 102, Name: Laptop, Price: 55000, Type: Electronics
ID: 201, Name: T-Shirt, Price: 499, Type: Clothing
ID: 202, Name: Jeans, Price: 1499, Type: Clothing
ID: 205, Name: Bolt speaker, Price: 3000, Type: Electronics
ID: 206, Name: HP mouse, Price: 4000, Type: Electronics


=== Order Management System ===
1. Create User
2. Create Product
3. Create Order
4. Cancel Order
5. Get All Products
6. Get Orders by User
7. Exit
Enter your choice: 6
Enter User ID to fetch orders: 3
Orders by User ID 3:
Product ID: 101, Name: Smartphone, Type: Electronics
Product ID: 202, Name: Jeans, Type: Clothing
=== Order Management System ===
1. Create User
2. Create Product
3. Create Order
4. Cancel Order
5. Get All Products
6. Get Orders by User
7. Exit
Enter your choice: 7
Exiting Order Management System.