Project: Layers Bakeshop

Session 2023_2027

Submitted By:

Amna Khalid 2023_CS_163

Supervised by:

Sir Laeeq Khan Niazi

Course:

CSC-103 Object Oriented Programming

Department of Computer Science

University of Engineering and Technology

Lahore Pakistan

Contents

Description:

Layers bakeshop system is a specialized software solution for managing bakery operations efficiently. Layers bakeshop system simplifies bakery management w manages inventory, recipes, and staff scheduling. Facilitates easy ordering, loyalty programs, and personalized service. Handles product availability, pricing, and sales transactions.

User Functionalities:

This system supports two users.

i) Admin ii) Customers

Each role has distinct functionalities.

Admin:

i. Admin can add products.
ii. Amin can remove products.
iii. Admin can view stock.
iv. Admin can edit the data of products.
v. Admin can change the password of his/her profile.
vi. He/she can edit the quantity of products.

Customer:

i. A customer can enter his/her profile.

ii. He/she can change the password of profile.
iii. Customer can view stock.
iv. He can add products to cart.
v. He can view his total purchase.
vi. A customer can confirm order.

Wireframes:

Figure 1: Welcome Page

Figure 2: Main Menu Page

Figure 3: Sign Up Page

Figure 4: Sign In Page

Figure 5: Admin Menu

Figure 6: Customer Menu

Functional Requirements:

Users	Functions	Results
Admin	**1.** Add any product **2.** Delete any product **3.** Check the stock **4.** Change Password. **5.** Check quantity of the products. **6.** Edit the prices of the products.	**1.** Stock will be increased **2.** It will remove defected products **3.** Stock will be presented. **4.** To secure the account **5.** To manage the stock. **6.** Edit prices according to the market rate.
Customer	**1.** View products. **2.** Add products to cart	**1.** To see which products are available for sale.

	3. Change password. **4.** View total amount of cart **5.** Confirm order **6.** Can view information about the store.	**2.** Products he wants to purchase. **3.** To ensure security. **4.** To check how much the products worth **5.** To buy the products **6.** To be acknowledged about that.

Complete code of business application:

ProductDL(database):

```
public bool AddProduct(Product product)
    {
        SqlConnection sqlConnection = new SqlConnection(connection);
        sqlConnection.Open();
        SqlCommand sql = new SqlCommand("INSERT INTO Products (Name, Category, Quantity, Price) VALUES (@name, @category, @quantity, @price)", sqlConnection);
        sql.Parameters.AddWithValue("@name", product.GetName());
        sql.Parameters.AddWithValue("@category", product.GetCategory());
        sql.Parameters.AddWithValue("@quantity", product.GetQuantity());
        sql.Parameters.AddWithValue("@price", product.GetPrice());
        int i = sql.ExecuteNonQuery();
        sqlConnection.Close();
```

```
        return i > 0;

    public static bool DeleteProduct(Product product)
    {

        SqlConnection sqlConnection = new SqlConnection(connection);
        sqlConnection.Open();

        SqlCommand sql = new SqlCommand($"select Quantity from Products where
Name='{product.GetName()}'", sqlConnection);
        int OldQuantity = int.Parse(Convert.ToString(sql.ExecuteScalar()));

        SqlCommand sqlCommand = new SqlCommand($"update Products set Quantity=@add where
Name='{product.GetName()}'", sqlConnection);
        sqlCommand.Parameters.AddWithValue("@add", OldQuantity - product.GetQuantity());
        int i = sqlCommand.ExecuteNonQuery();

        sqlConnection.Close();

        return i > 0;
    }

    public static DataTable GetCart()
    {
        try
        {
            using (SqlConnection sqlConnection = new SqlConnection(connection))
            {
                sqlConnection.Open();
```

```
            string query = "SELECT * FROM Cart";
            SqlDataAdapter adapter = new SqlDataAdapter(query, sqlConnection);
            DataTable cartTable = new DataTable();
            adapter.Fill(cartTable);

            return cartTable;
          }
        }
        catch (Exception ex)
        {

          Console.WriteLine($"An error occurred: {ex.Message}");
          return null;
        }
      }

      public bool UpdatePrice(Product product)
      {

        SqlConnection sqlConnection = new SqlConnection(connection);
        sqlConnection.Open();
        SqlCommand sql = new SqlCommand("update Products set Price=@newprice where
Name=@name", sqlConnection);
        sql.Parameters.AddWithValue("@newprice", product.GetPrice());
        sql.Parameters.AddWithValue("@name", product.GetName());
        int i = sql.ExecuteNonQuery();
        sqlConnection.Close();
        return i > 0;
      }
```

ProductDL(file handling):

```
namespace ProjectDLL.DL.FH
{
   public class ProductFH:IProductDL
   {
      private static List<Product> products = new List<Product>();

      public ProductFH()
      {
         if (ReadFromFile())
         {
            Console.WriteLine();
         }
         else
         {
            Console.WriteLine("Error");
         }
      }

      public List<Product> GetAllProducts()
      {
         return products;
      }
      public bool AddProduct(Product product)
      {
         products.Add(product);
```

```
        StreamWriter streamWriter = new StreamWriter("Products.txt", true);

streamWriter.WriteLine($"{product.GetName()},{product.GetCategory()},{product.GetQuantity()},{produc
t.GetPrice()}");
        streamWriter.Flush();
        streamWriter.Close();
        WriteToFile();
        return true;
    }

    public bool DeleteWholeProduct(Product product)
    {
        foreach(Product p in products)
        {
            if (p.GetName() == product.GetName())
            {
                products.Remove(p);
                WriteToFile();

                return true;
            }
        }
        return false;
    }

    public List<string> GetProductNames()
    {
        List<string> names = new List<string>();
        foreach (Product product in products)
```

```
        {
            names.Add(product.GetName());
        }
        return names;
    }

    public bool UpdatePrice(Product product)
    {
        foreach(Product p in products)
        {
            if (p.GetName() == product.GetName())
            {
                p.SetPrice(product.GetPrice());

                WriteToFile();
            }
        }
        return true;
    }

    private bool ReadFromFile()
    {

        StreamReader streamReader = new StreamReader("Products.txt");
        string record;

        if (File.Exists("Products.txt"))
        {
            while ((record = streamReader.ReadLine()) != null)
```

```
            {
                string[] splittedRecord = record.Split(',');
                string Name = splittedRecord[0];
                string Category = splittedRecord[1];
                int Quantity = int.Parse(splittedRecord[2]);
                double Price = double.Parse(splittedRecord[3]);

                Product product = new Product(Name,Category,Quantity,Price);
                products.Add(product);
            }
            streamReader.Close();
            return true;
        }
        else
        {
            return false;
        }
    }

    private void WriteToFile()
    {

        StreamWriter streamWriter = new StreamWriter("Products.txt");

        foreach (Product product in products)
        {

streamWriter.WriteLine($"{product.GetName()},{product.GetCategory()},{product.GetQuantity()},{produc
t.GetPrice()}");
```

```
        }


        streamWriter.Close();
      }
   }
}
```

ProductBL:

```
namespace ProjectDLL.BL
{
   public class Product
   {
      private string Name;
      private string Category;
      private int Quantity;
      private double Price;




      public Product(string name, string category, int quantity, double price)
      {
         Name = name;
         Category = category;
         Quantity = quantity;
         Price = price;
      }
      public Product() { }
```

```
    public Product(string name, int quantity)
    {
       Name = name;
       Quantity = quantity;
    }
    public Product(string name)
    {
       Name = name;
    }
    public Product(string name, double price)
    {
       Name = name;
       Price = price;
    }
    public Product(Product p)
    {
       Name = p.Name;
       Category = p.Category;
       Quantity = p.Quantity;
       Price = p.Price;
    }

    public void SetName(string name)
    {
       Name = name;
    }

    public void SetCategory(string category)
    {
```

```
        Category = category;
    }

    public string GetCategory()
    {
        return Category;
    }

    public int GetQuantity()
    {
        return Quantity;
    }

    public void SetQuantity(int quantity)
    {
        Quantity = quantity;
    }

    public void SetPrice(double price)
    {
        Price = price;
    }

    public double GetPrice()
    {
        return Price;
    }

    public string GetName()
```

```
        {
            return Name;
        }
    }
}
```

ConsoleUI:

```
namespace ConsoleProject.UI
{
    internal class ProductUI
    {

        public static string MainMenu()
        {
            Console.WriteLine("\n1. View Products");
            Console.WriteLine("2. Add New Product");
            Console.WriteLine("3. Delete Product");
            Console.WriteLine("4. Update Product Price");
            Console.WriteLine("5. Exit\n");

            Console.Write("Your option is .... ");
            string option = Console.ReadLine();

            return option;
        }

        public static string ViewProducts()
        {
```

```
        List<Product> products = ObjectHandler.GetProductDL().GetAllProducts();

        if (products.Count != 0)
        {
            foreach (Product product in products)
            {
                Console.WriteLine($"Name: {product.GetName()} \t Category: {product.GetCategory()} \t
Quantity: {product.GetQuantity()} \t Price: {product.GetPrice()}\n");
            }
            return "Current Products";
        }
        else
        {
            return "No Current Products";
        }
    }

    public static string AddProduct()
    {
        while (true)
        {
            try
            {
                Console.Write("Enter Name Of Product: ");
                string name = Console.ReadLine();

                Console.Write("Enter Category: ");
                string category = Console.ReadLine();
```

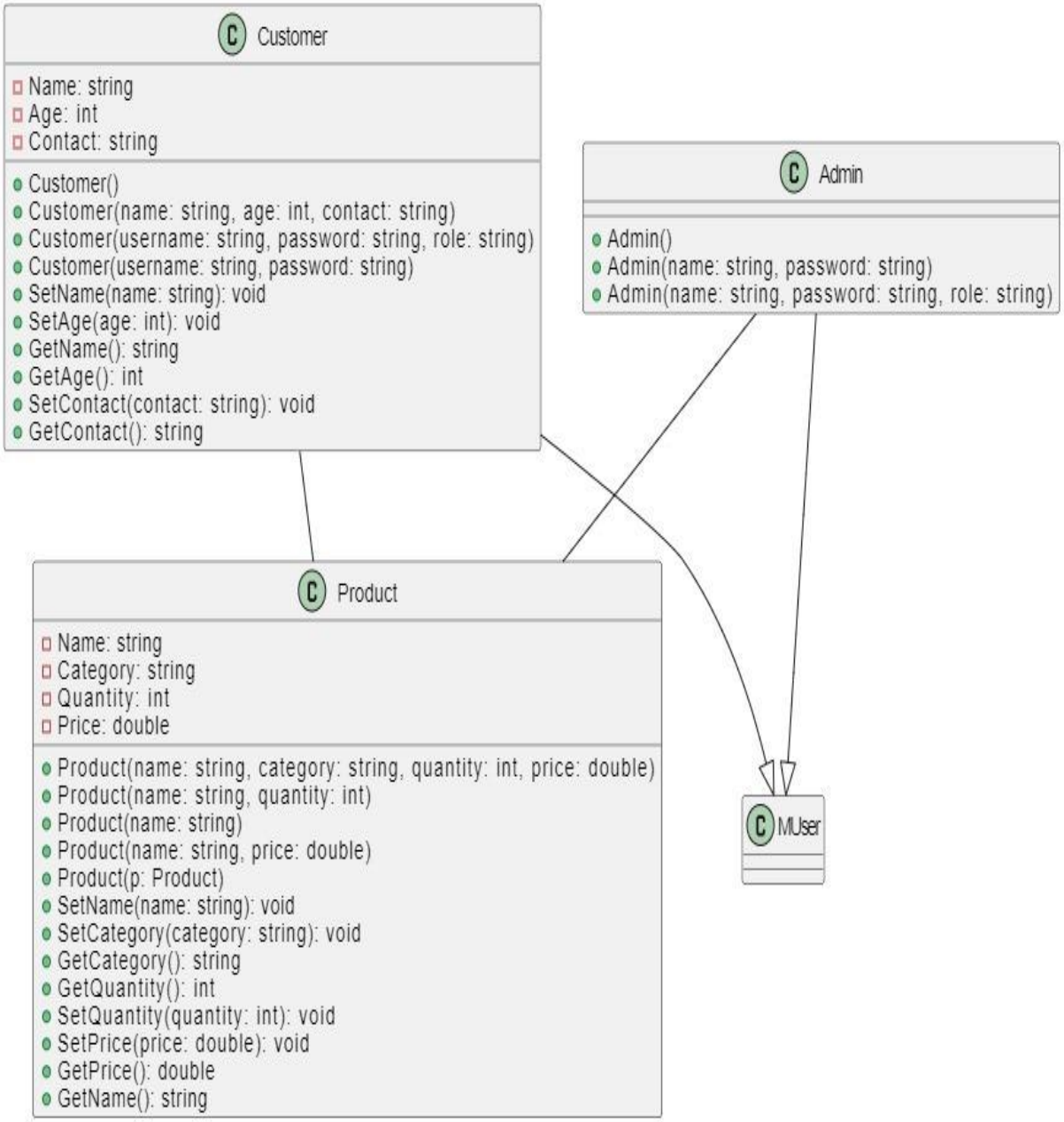
```
            Console.Write("Enter Total Quantity Of The Product: ");
            int quantity = int.Parse(Console.ReadLine());

            Console.Write("Enter Price Of The Product: ");
            double price = double.Parse(Console.ReadLine());

            Console.WriteLine();

            Product product = new Product(name,category,quantity,price);

            if (ObjectHandler.GetProductDL().AddProduct(product))
            {
                return "Product Added Successfully!";
            }
            else
            {
                return "Error 404!";
            }
        }
        catch
        {
            Console.WriteLine("\nEnter Valid Input!");
            Console.WriteLine("\nPress Any Key to Continue..");
            Console.ReadKey();
            Console.Clear();

        }
    }
```

```
    }

    public static string DeleteProduct()
    {
       while (true)
       {
          List<string> Products = ObjectHandler.GetProductDL().GetProductNames();

          foreach (string Product in Products)
          {
             Console.WriteLine($"{Product}");

          }

          Console.Write("\nEnter Name Of Product You Want To Delete: ");
          string name = Console.ReadLine();

          Product product = new Product(name);


          if (ObjectHandler.GetProductDL().DeleteWholeProduct(product))
          {
             return "Product Removed!";
          }
          else
          {
             Console.WriteLine("Enter Correct Product Name!");
             Console.WriteLine("\nPress Any Key to Continue..");
             Console.ReadKey();
```

```
            Console.Clear();
        }

    }
}


public static string UpdatePrice()
{
    while (true)
    {
        try
        {
            List<string> Products = ObjectHandler.GetProductDL().GetProductNames();

            foreach (string Product in Products)
            {
                Console.WriteLine($"{Product}");

            }

            Console.Write("\nEnter Name Of Product You Want To Update Price Of: ");
            string name = Console.ReadLine();

            Console.Write("Enter New Price: ");
            double price  = double.Parse(Console.ReadLine());

            Product product = new Product(name, price);
```

```
                if (ObjectHandler.GetProductDL().UpdatePrice(product))
                {
                    return "Product Price Updated!";
                }
                else
                {
                    return "Error!";
                }
            }
            catch
            {
                Console.WriteLine("\nEnter Valid Input!");
                Console.WriteLine("\nPress Any Key to Continue..");
                Console.ReadKey();
                Console.Clear();
            }
        }
    }
  }
}
```

CRC:


Customer
Name: string
Age: int
Contact: string
Customer()
Customer(name: string, age: int, contact: string)
Customer(username: string, password: string, role: string)
Customer(username: string, password: string)
SetName(name: string): void
SetAge(age: int): void
GetName(): string
GetAge(): int
SetContact(contact: string): void
GetContact(): string
Admin
Admin()
Admin(name: string, password: string)
Admin(name: string, password: string, role: string)
Product
Name: string
Category: string
Quantity: int
Price: double
Product(name: string, category: string, quantity: int, price: double)
Product(name: string, quantity: int)
Product(name: string)
Product(name: string, price: double)
Product(p: Product)
SetName(name: string): void
SetCategory(category: string): void
GetCategory(): string
GetQuantity(): int
SetQuantity(quantity: int): void
SetPrice(price: double): void
GetPrice(): double
GetName(): string
MUser