**Pharmacy Management System**

****

Session: 2022 – 2026

**Submitted by:**

Muhammad Taha Saleem 2022-CS-139

**Supervised by:**

Prof. Dr. Muhammad Awais Hassan

Department of Computer Science

**University of Engineering and Technology**

**Lahore Pakistan**

**Table of Contents**

**Introduction………………………………………………………………….…………...……3**

**Class Responsibility Collaboration Card……………………………...………..……………4**

**Object Oriented Programming……………………………………………..….…………...…5**

**Design Pattern Implementation..…………………………………………………….……...7,8**

**Class Details……........……………………………………………………………..….......….8,9**

**Conclusion……........……………………………………………………………..……........…..9**

**Introduction**

* **Overview**

Pharmacy Management System is a backend management application developed in C#, catering to two types of users: Admin and Pharmacist. It utilizes object-oriented programming concepts to organize and manage various tasks within a Pharmacy. The system provides efficient management and operational support, enabling seamless workflow and outclass customer service.

* **YouTube Link**

[**https://youtu.be/Hq-GOm0SpZ0**](https://youtu.be/Hq-GOm0SpZ0)

* **Objectives**

The objectives of this system includes

1. **Efficient Backend Management**: The system aims to streamline backend operations by providing tools and functionalities to effectively manage medicines, customers, and sales data.
2. **Improved Customer Service**: By maintaining users records, processing orders accurately, and offering support, the system aims to enhance customer satisfaction and provide personalized services.

* **Functionality**

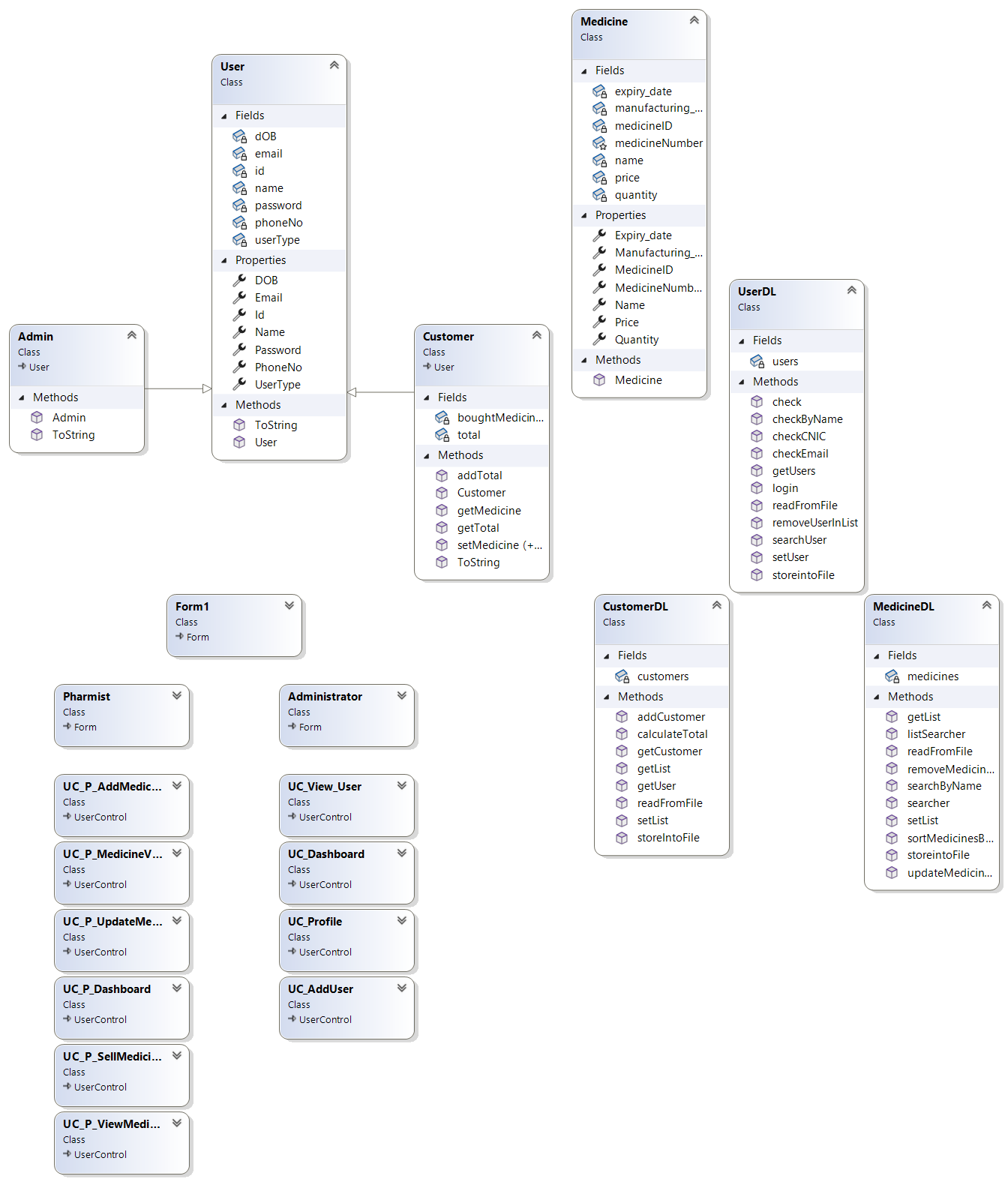
**Admin:**

1. **Medicine Management:** Add, edit, or remove medicine and beverage items, set prices, manage quantities, and update attributes.
2. **Users Management:** Add, edit, or remove admins and pharmacists, assign roles, update personal information, and track performance.
3. **Expiry Management:** Tracking Medicines validity record.

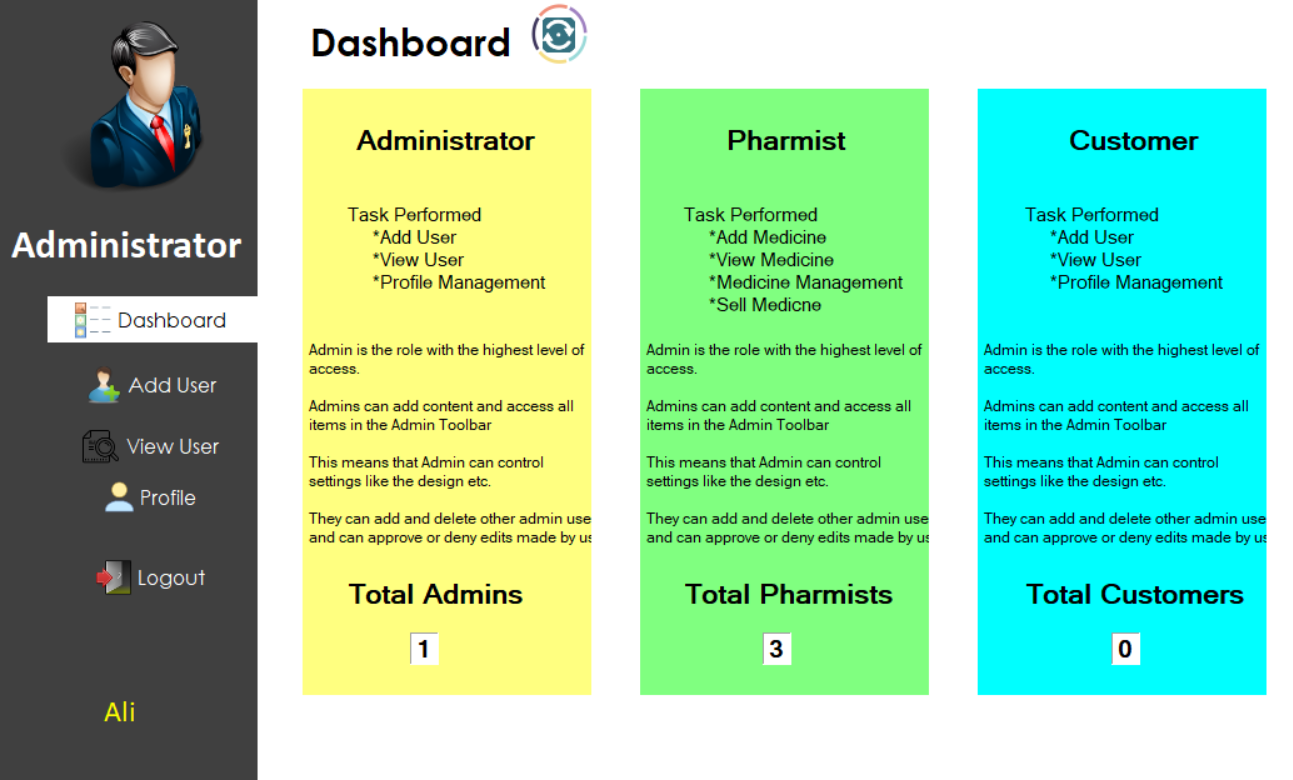
**Customer:**

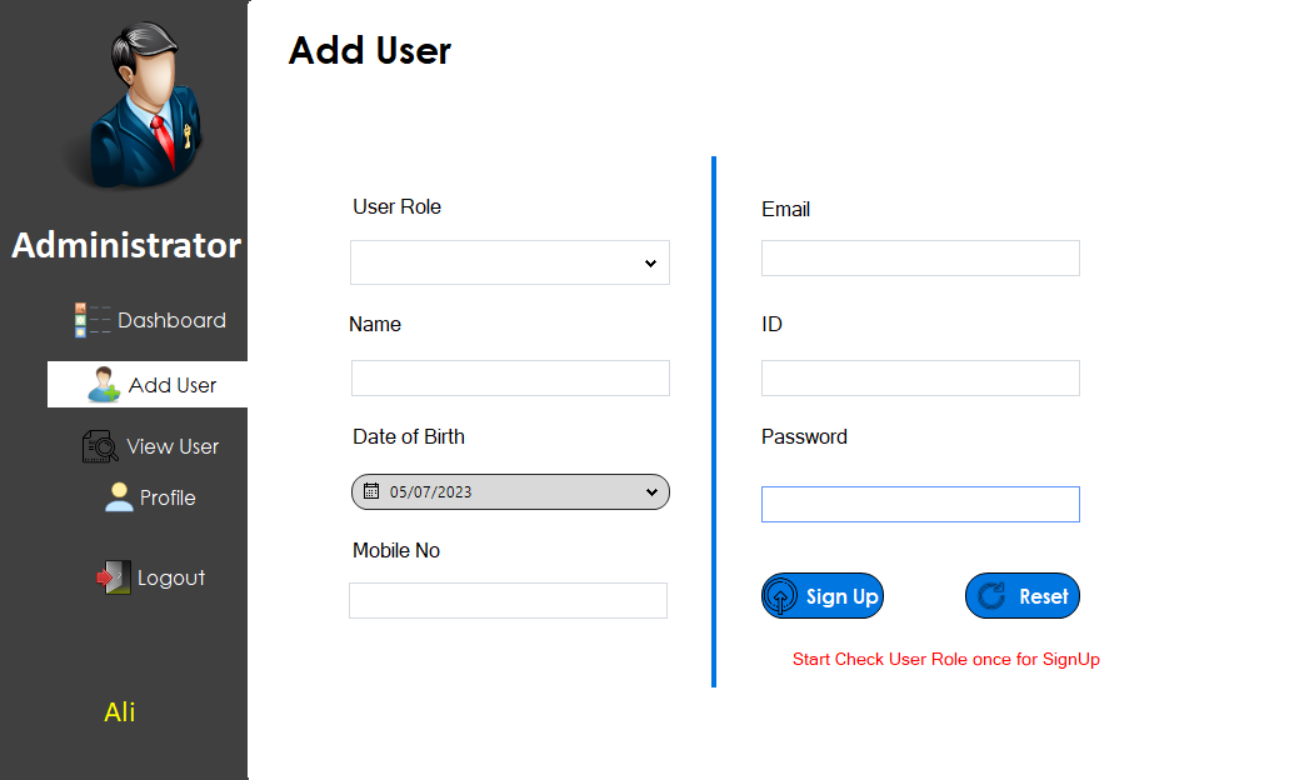
1. **Buying Process:** Buying medicines, calculate total amount, and generate invoices or receipts.

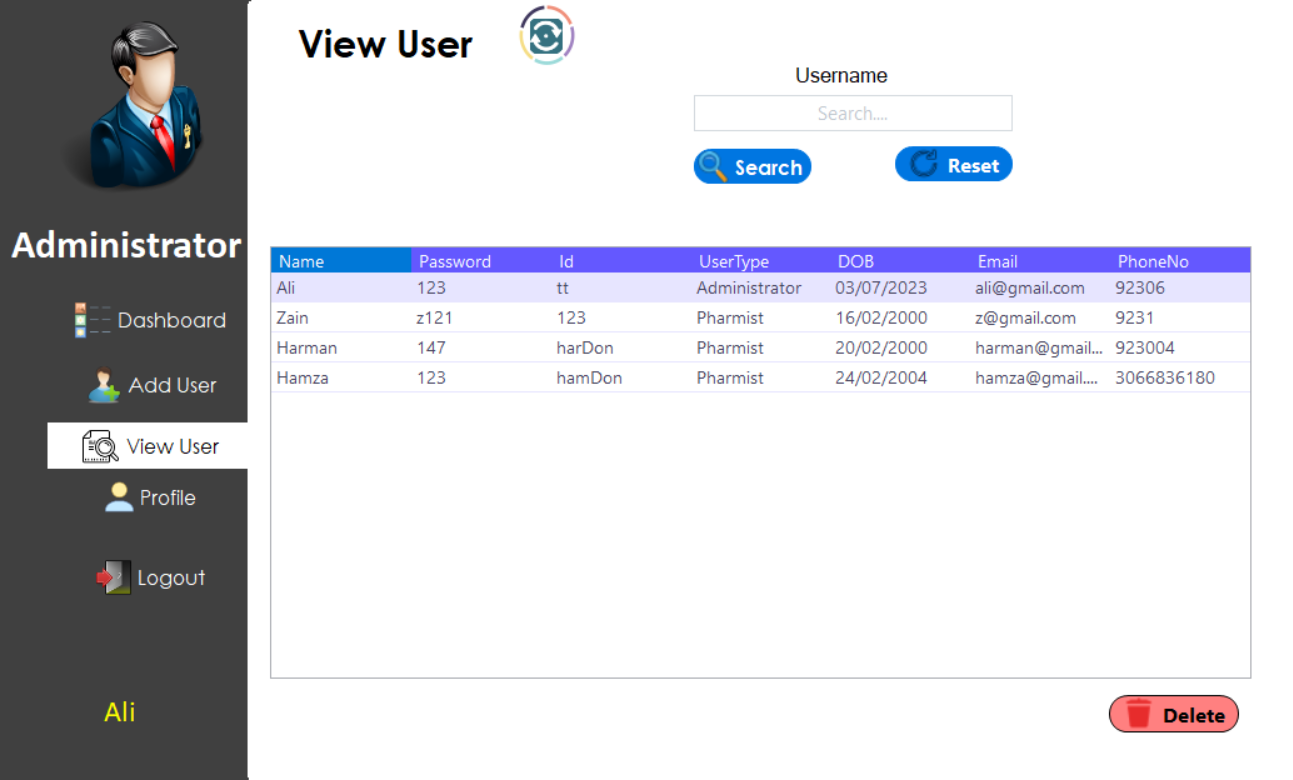
**Class Responsibility Collaboration Diagrams**

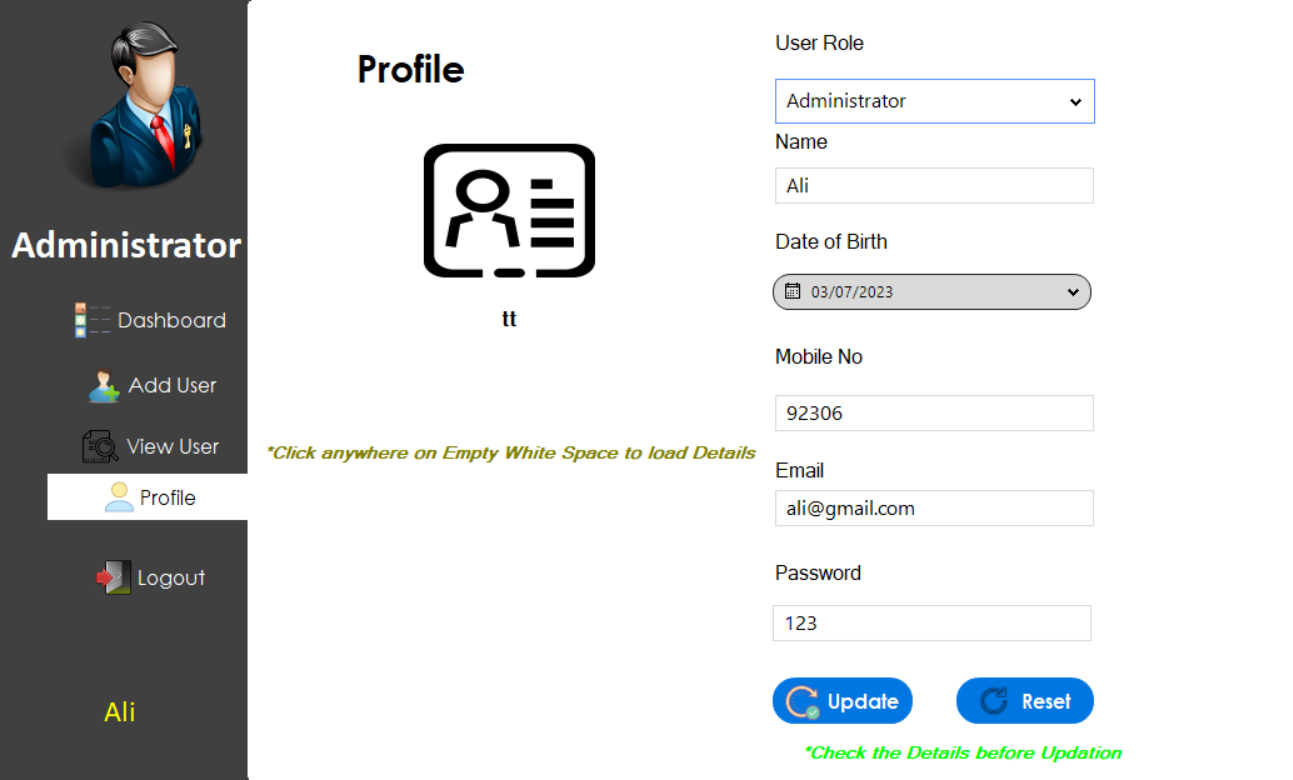
****

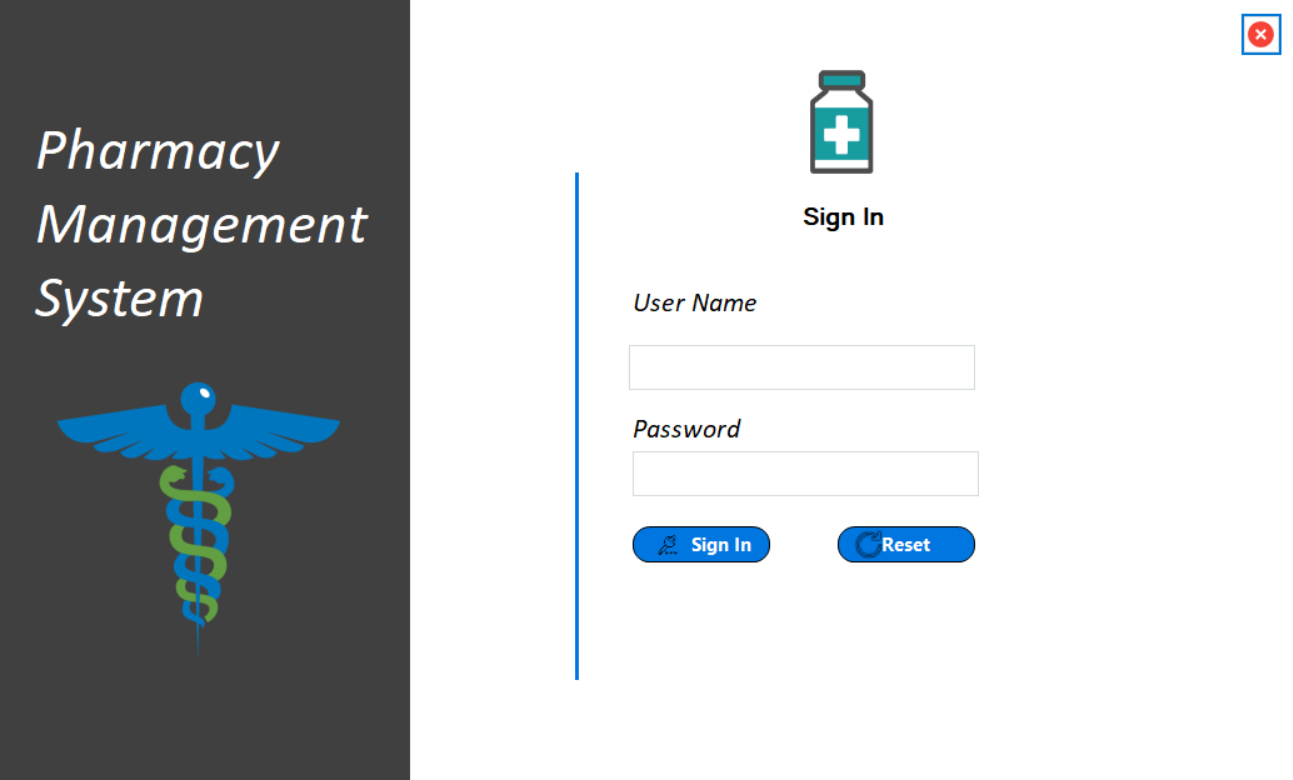
**Wire Frames**

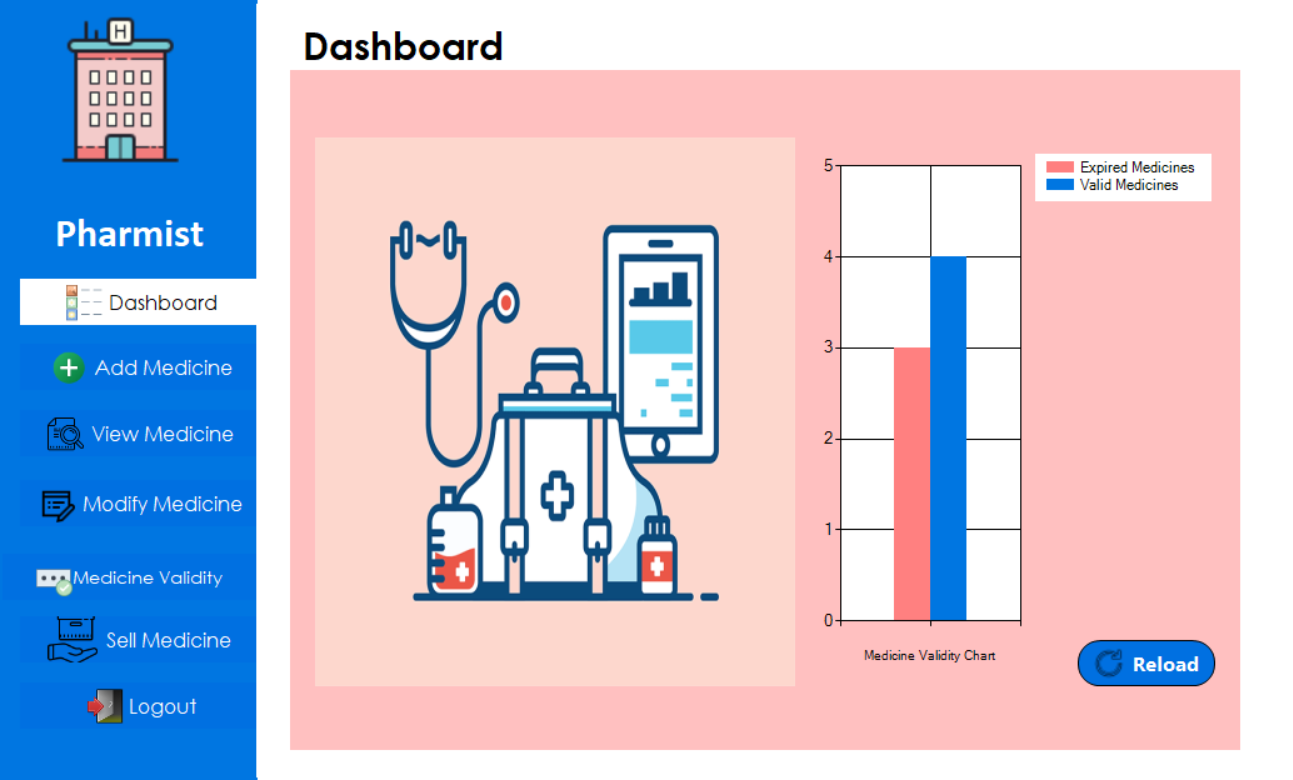
****

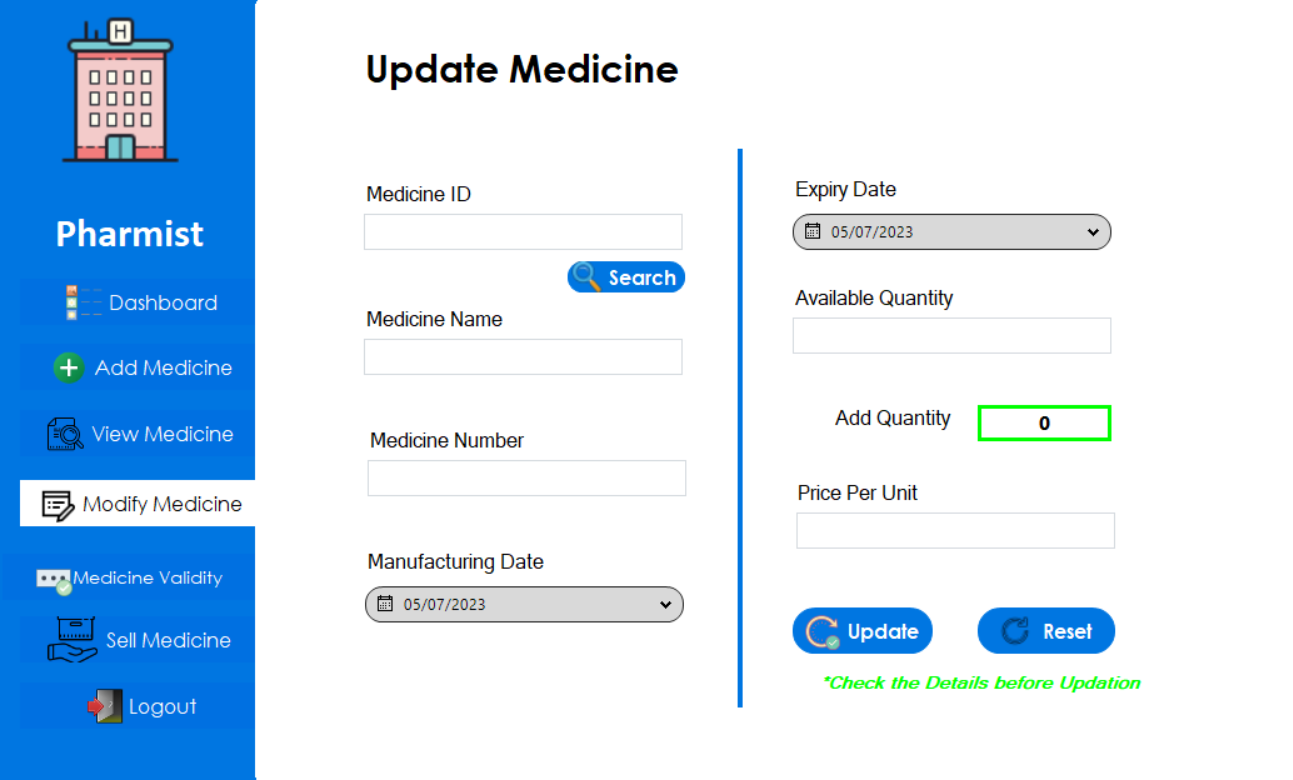
****

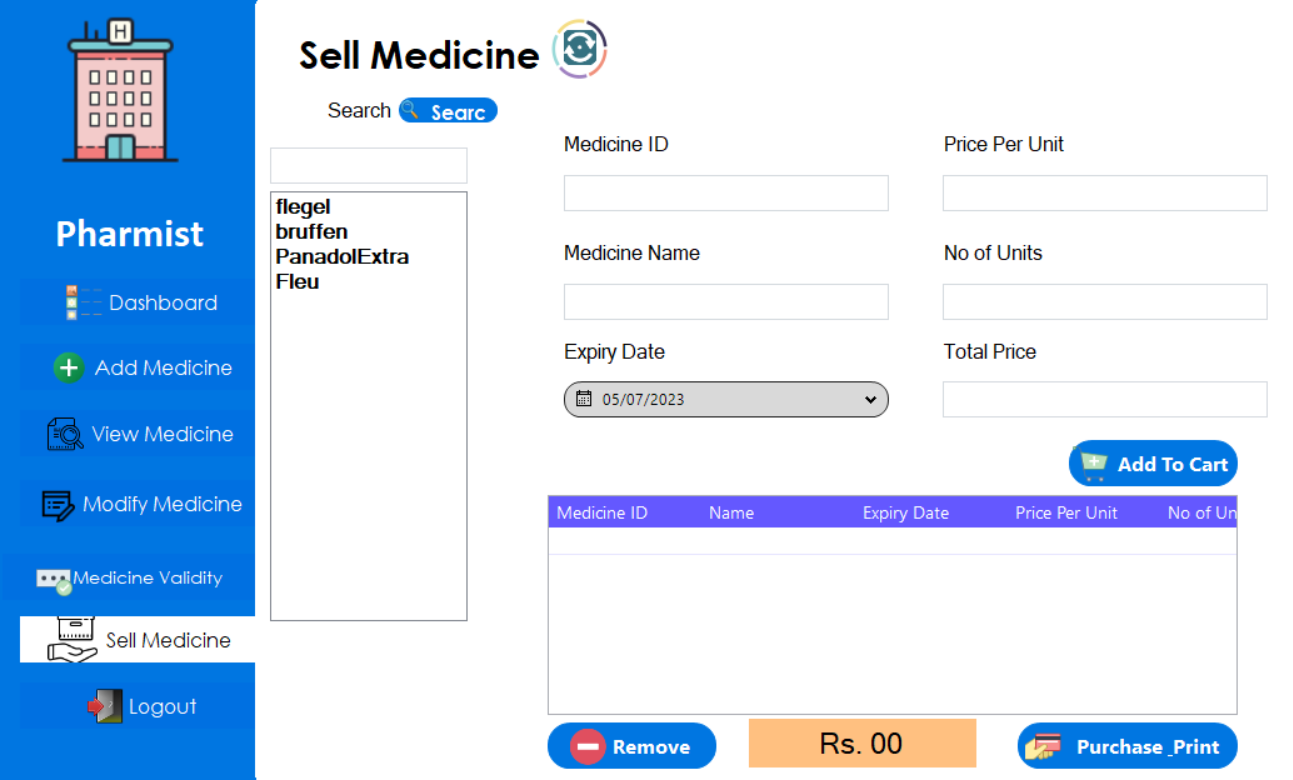
****

****

****

****

****

****

**Object Oriented Programming**

Object-oriented programming (OOP) is a programming paradigm based on the concepts of the “objects”, which contains data and code called as attributes and behavior of the class respectively. The main concepts of OOP includes association, inheritance and polymorphism. I have used this programming paradigm in my project.

* **Association**

In Object-oriented programming, Association is a relation between two separate classes which establishes through their objects. Association can be one-to-one, one-to-many, many-to-one, many -to-many. There are two types of Association, **Aggregation** and **Composition**. I have used both Composition & Aggregation in my project. It is used in three places.

1. Pharmacist can add multiple medicines which customer wants to buy. Medicine can exist without the object of Pharmacist, so I used aggregation for this particular purpose.
2. Admin can add many Pharmacists as well as Admins for his shop, in this case they (Pharmacists and Admins) can exist without Admin so it is aggregation also.

**Advantage**

If I compare this with my procedural programming concepts, I can observe that there is a clear advantage of OOP. There was disjoint data of Medicine and Customer in procedural programming which is rectified in OOP. Now the lists of Customer and bought Medicine are within the class of Customer.

* **Inheritance**

Inheritance is one of the core concepts of Object-oriented programming approach. It is a feature that allows a new class to derive from an existing class. The new class inherits all the public or protected attributes and the member functions of the base class. I have used this OOP concept in one place in my management system.

1. User class is a parent class. Admin and Customer class are derived from this class. Admin and Customer are two users. So, that is the reason I have applied the concept of inheritance here as they are inheriting user object which contains name, password, role, DOB, cell no and id form the User class.

**Advantage**

Inheritance gives various advantages over procedural programming. It promotes code-reusability and reduces redundancy. It helps in organizing the program’s structure. It allows flexibility in the code as you will adjust in one place and the rest of the code will work smoothly.

* **Polymorphism**

Polymorphism is also one of the core concepts of Object-oriented programming approach. This concept refers to the ability of a function to perform multiple operation under different circumstances. There are two types of Polymorphism. The type of polymorphism used to extend the functionality of common functions in parent and child classes is called Dynamic Polymorphism. I have used Dynamic Polymorphism in few places.

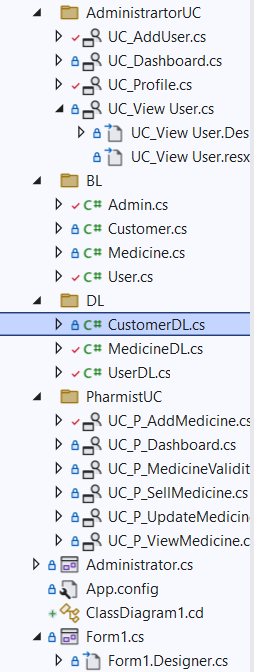
1. toString() functions in inheritance (User – Customer & Admin).This function in parent class of User will return the details of object containing at the time it is called in parent child classes because it is also overridden in child classes.
2. I have also used this type of Polymorphism in menu() function. This function would load the interface of the user according to their role when logged in correctly.

The other type of Polymorphism is Static Polymorphism. I have used this polymorphism for the constructors and login() and other search functions.

**Advantage**

Polymorphism allowed us to extend the functionalities from the base class to use them for the child classes. Due to polymorphism, our code has become shorter because it didn’t required us to copy paste the whole code and then make changes to it. Dynamic Polymorphism has enabled the programmer to use the same function in different manner. We lacked this when we were making our projects in procedural programming.

* **Design Pattern Implementation**

The directory structure for the project is given below: 

* **Business Logic (BL)**

The Business Logic folder contains the main classes of the project. It includes the classes of Users (Credentials), Employee, Admin, Supplier, Medicine, and Customer, Feedbacks.

* **Data Layer (DL)**

The Data Layer folder contains the static Lists and functions of the project. It contains the list of medicines, list of users which contains the objects of Admin and Customer. There are static functions of each class as well such as storing and loading of data to and from files. It also includes the other static functions.

* **User Interface or User Control (UI/UC)**

This folder contains all the code for Forms for interaction with user.

**Class Details**

* **User**

A User class is used for generalizing the two classes of Admin and Customer. It is a parent class of these two classes. It contains the common attributes of the both classes. User class contains strings of username, password and role of the user. Its attributes are also private. This class is created to facilitate in the sign-in and sign-up procedure. There are two constructors in this class. One constructor with three arguments are used when sign-up functionality is used. The other constructor with two arguments is used when we use sign-in function. There is another constructor which takes four arguments including id. So, that cashier can only login through his/her id once signing up.

* **Admin**

This class inherits the attributes and behavior of the parent class User.

* **Customer/Pharmacist**

Customer class is a child class of User class. It inherits the attributes and behavior of the parent class. All the attributes of the Customer class are also private. Its attributes include list of bought medicines and function to add the object of medicine to the defined list as customer is able to buy medicines. The Customer class has aggregation relation with Medicine class.

* **Medicine**

Medicine class contains the attributes of medicine name, medicine price, and medicine expiry date, medicine category. This class has relation of aggregation with Customer or Pharmacist as the customer class contains the list of products and pharmacist is able to add product object to its list.

* **Customer/Pharmacist**

The Customer class is a containing the attributes like customer name, customer id, total, and list of orders which in fact is the list of medicines as the customer can order multiple medicines. The customer class has one to many relation with medicine class.

**Conclusion**

In conclusion, the Pharmacy Management System is built using the object-oriented programming approach. Its key functionalities includes the CRUD operations. Important concepts object-oriented concepts such as association, inheritance and polymorphism are used in this system. I faced several challenges during this phase. I faced difficulty in designing an effective class diagram collaboration model for the system and managing the key concepts of OOP paradigm. Throughout the period of designing, production and development of this project, I have learned how to create an effective system using object-oriented theory. The object-oriented approach can be really helpful in scaling of the project. It also helps the programmers in future to maintain the software.

**Code:**

**BL:**

**User Class:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Pharmacy\_Management\_System.BL

{

public class User

{

private string name = "";

private string password = "";

private string id = "";

private string userType = "";

private DateTime dOB;

private string email = "";

private long phoneNo;

public string Name { get => name; set => name = value; }

public string Password { get => password; set => password = value; }

public string Id { get => id; set => id = value; }

public string UserType { get => userType; set => userType = value; }

public DateTime DOB { get => dOB; set => dOB = value; }

public string Email { get => email; set => email = value; }

public long PhoneNo { get => phoneNo; set => phoneNo = value; }

public User(string userType,string name, DateTime dOB, long phoneNo, string email, string password, string id)

{

this.id = id;

this.name = name;

this.password = password;

this.userType = userType;

DOB = dOB;

this.email = email;

this.phoneNo = phoneNo;

}

public virtual string ToString()

{

return "Admin Details : \n Name: " + name + " Password: " + password + " ID: " + id + " UserType: " + userType + " DOB: "+ DOB + " Contact: " + phoneNo + " Email: " + email;

}

}

}

**Admin Class:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Xml.Linq;

namespace Pharmacy\_Management\_System.BL

{

public class Admin : User

{

public Admin(string name, string password, string id, string userType, DateTime dob, string email, long phoneNo) : base(userType,name,dob,phoneNo,email,password,id)

{

}

public override string ToString()

{

return "Admin Details : \n Name: " + this.Name + " Password: " + this.Password + " ID: " + this.Id + " UserType: " + this.UserType + " DOB: " + this.DOB + " Contact: " + this.PhoneNo + " Email: " + this.Email;

}

}

}

**Customer/Pharmacist Class:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Pharmacy\_Management\_System.BL

{

public class Customer : User

{

private float total;

private List<Medicine> boughtMedicines = new List<Medicine>();

public Customer(string name, string password, string id, string userType,DateTime dob,string email, Int64 phoneNo) : base(userType, name, dob, phoneNo, email, password, id)

{

this.total = 0;

}

public void addTotal(float total)

{

//this.total += total;

this.total = total;

}

public float getTotal()

{

return this.total;

}

public void setMedicine(Medicine boughtMedicine)

{

boughtMedicines.Add(boughtMedicine);

}

public List<Medicine> getMedicine()

{

return this.boughtMedicines;

}

public void setMedicine()

{

boughtMedicines.Clear();

}

public void setMedicine(string name)

{

for (int i = 0; i < boughtMedicines.Count; i++)

{

if (boughtMedicines[i].Name == name)

{

boughtMedicines.RemoveAt(i);

break;

}

}

}

public override string ToString()

{

return "Admin Details : \n Name: " + this.Name + " Password: " + this.Password + " ID: " + this.Id + " UserType: " + this.UserType + " DOB: " + this.DOB + " Contact: " + this.PhoneNo + " Email: " + this.Email;

}

}

}

**Medicine Class:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Pharmacy\_Management\_System.BL

{

public class Medicine

{

private string medicineID = "";

private string name = "";

protected string medicineNumber = "";

private DateTime manufacturing\_date;

private DateTime expiry\_date;

private int quantity;

private float price;

public string MedicineID { get => medicineID; set => medicineID = value; }

public string Name { get => name; set => name = value; }

public string MedicineNumber { get => medicineNumber; set => medicineNumber = value; }

public DateTime Expiry\_date { get => expiry\_date; set => expiry\_date = value; }

public DateTime Manufacturing\_date { get => manufacturing\_date; set => manufacturing\_date = value; }

public int Quantity { get => quantity; set => quantity = value; }

public float Price { get => price; set => price = value; }

public Medicine(string medicineID,string name,string medicineNumber,DateTime manufacturing\_date,DateTime expiry\_date,int Quantity,float price)

{

this.medicineID = medicineID;

this.name = name;

this.medicineNumber = medicineNumber;

this.manufacturing\_date = manufacturing\_date;

this.expiry\_date = expiry\_date;

this.quantity = Quantity;

this.price = price;

}

}

}

**DL:**

**MedicineDL Class:**

using Pharmacy\_Management\_System.BL;

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Pharmacy\_Management\_System.DL

{

public class MedicineDL

{

private static List<Medicine> medicines = new List<Medicine>();

public static void setList(Medicine medicine)

{

medicines.Add(medicine);

}

public static List<Medicine> getList()

{

return medicines;

}

public static bool updateMedicineInList(string name, float price)

{

foreach (Medicine item in MedicineDL.getList())

{

if (name == item.Name)

{

item.Price = price;

return true;

}

}

return false;

}

public static bool updateMedicineInList(string name, DateTime expiry)

{

foreach (Medicine item in MedicineDL.getList())

{

if (name == item.Name)

{

item.Expiry\_date = expiry;

return true;

}

}

return false;

}

public static bool removeMedicineInList(string ID)

{

for (int i = 0; i < MedicineDL.getList().Count; i++)

{

if (MedicineDL.getList()[i].MedicineID == ID)

{

MedicineDL.getList().RemoveAt(i);

return true;

}

}

return false;

}

public static List<Medicine> sortMedicinesByPrice()

{

List<Medicine> sortedList = MedicineDL.getList().OrderBy(o => o.Price).ToList();

return sortedList;

}

public static Medicine searcher(string id)

{

foreach (Medicine item in MedicineDL.getList())

{

if (id == item.MedicineID)

{

return item;

}

}

return null;

}

public static List<Medicine> listSearcher(string name)

{

List<Medicine> medicines = new List<Medicine>();

lock (MedicineDL.getList())

{

foreach (Medicine item in MedicineDL.getList())

{

if (item.Name == name)

{

medicines.Add(item);

}

}

}

return medicines;

}

public static Medicine searchByName(string name)

{

foreach (Medicine item in MedicineDL.getList())

{

if (name == item.Name)

{

return item;

}

}

return null;

}

public static bool readFromFile(string filePath)

{

StreamReader f = new StreamReader(filePath);

string record;

if (File.Exists(filePath))

{

while ((record = f.ReadLine()) != null)

{

string[] splittedRecord = record.Split(',');

string medicineID = splittedRecord[0];

string medicineName = splittedRecord[1];

string medicineNumber = splittedRecord[2];

DateTime manufacturingDate = DateTime.Parse(splittedRecord[3]);

DateTime medicineExpiry = DateTime.Parse(splittedRecord[4]);

int quantity = int.Parse(splittedRecord[5]);

float medicinePrice = float.Parse(splittedRecord[6]);

Medicine medicine = new Medicine(medicineID, medicineName, medicineNumber,manufacturingDate, medicineExpiry, quantity,medicinePrice);

setList(medicine);

}

f.Close();

return true;

}

else

{

return false;

}

}

public static void storeintoFile(string filePath)

{

StreamWriter f = new StreamWriter(filePath);

foreach (Medicine medicine in MedicineDL.getList())

{

f.WriteLine(medicine.MedicineID + "," + medicine.Name + "," + medicine.MedicineNumber + "," + medicine.Manufacturing\_date + "," + medicine.Expiry\_date + "," + medicine.Quantity + "," + medicine.Price );

}

f.Flush();

f.Close();

}

}

}

**UserDL Class:**

using Pharmacy\_Management\_System.BL;

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Xml.Linq;

namespace Pharmacy\_Management\_System.DL

{

public class UserDL

{

private static List<User> users = new List<User>();

public static List<User> getUsers()

{

return users;

}

public static void setUser(User user)

{

users.Add(user);

}

public static List<User> searchUser(string name)

{

List<User> users = new List<User> ();

foreach (User item in UserDL.getUsers())

{

if (item.Name == name)

{

users.Add(item);

}

}

if (users.Count!=0)

{

return users;

}

return null;

}

public static User login(string username, string password)

{

foreach (User item in UserDL.getUsers())

{

if (item.Name == username && item.Password == password)

{

return item;

}

}

return null;

}

public static User check(string ID)

{

foreach (User item in UserDL.getUsers())

{

if (item.Id == ID)

{

return item;

}

}

return null;

}

public static User checkByName(string name)

{

foreach (User item in UserDL.getUsers())

{

if (item.Name == name)

{

return item;

}

}

return null;

}

public static bool readFromFile(string filePath)

{

StreamReader f = new StreamReader(filePath);

string record;

if (File.Exists(filePath))

{

while ((record = f.ReadLine()) != null)

{

string[] splittedRecord = record.Split(',');

string username = splittedRecord[0];

string password = splittedRecord[1];

string id = splittedRecord[2];

string userType = splittedRecord[3];

DateTime dob = DateTime.Parse(splittedRecord[4]);

string email = splittedRecord[5];

long phone = long.Parse(splittedRecord[6]);

User user = new User(userType, username, dob, phone, email, password, id);

setUser(user);

}

f.Close();

return true;

}

else

{

return false;

}

}

public static void storeintoFile(string filePath)

{

StreamWriter f = new StreamWriter(filePath);

foreach (User user in UserDL.getUsers())

{

f.WriteLine(user.Name + "," + user.Password + "," + user.Id + "," + user.UserType + "," + user.DOB + "," + user.Email + "," + user.PhoneNo);

}

f.Flush();

f.Close();

}

public static bool removeUserInList(string usernameOrID)

{

foreach (User item in UserDL.getUsers())

{

if (item.Id == usernameOrID)

{

UserDL.getUsers().Remove(item);

UserDL.storeintoFile(Program.userPath);

return true;

}

}

return false;

}

// Check Email Format......................................................

public static bool checkEmail(string email)

{

char[] Mail = { '@', 'g', 'm', 'a', 'i', 'l', '.', 'c', 'o', 'm' };

int lengthOfEmail = email.Length - 11;

int countEmail = email.Length - 1;

int i = 9;

bool flag = false;

while (countEmail != lengthOfEmail)

{

if (email[countEmail] == Mail[i] || email[0] != '@')

{

flag = true;

}

else

{

flag = false;

break;

}

countEmail--;

i--;

}

return flag;

}

//Check CNIC for Sign up ........................................

public static bool checkCNIC(long aCNIC)

{

long firstDigit = aCNIC / 1000000000000;

long cnic = aCNIC, b = 0;

while (cnic != 0)

{

cnic = cnic / 10;

b++;

if (cnic == 0)

{

if (b != 13 && firstDigit != 3)

{

return false;

}

else if (b == 13 && firstDigit == 3)

{

return true;

}

}

}

return false;

}

}

}

**CustomerDL Class:**

using Pharmacy\_Management\_System.BL;

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Pharmacy\_Management\_System.DL

{

public class CustomerDL

{

private static List<Customer> customers = new List<Customer>();

public static List<Customer> getList()

{

return customers;

}

public static void setList(Customer customer)

{

customers.Add(customer);

}

public static float calculateTotal()

{

float total = 0;

foreach (Customer item in customers)

{

for (int i = 0; i < item.getMedicine().Count; i++)

{

total = total + item.getMedicine()[i].Price;

item.addTotal(total);

}

}

return total;

}

public static User getUser(string id)

{

foreach (User item in UserDL.getUsers())

{

if (item.Id == id)

{

return item;

}

}

return null;

}

public static Customer addCustomer(string name, string password, string id, string userType, DateTime dob,string email,int phone)

{

Customer c = new Customer(name, password, id, userType,dob,email,phone);

return c;

}

public static Customer getCustomer(string id)

{

foreach (Customer item in customers)

{

if (id == item.Id)

{

return item;

}

}

return null;

}

public static bool readFromFile(string path)

{

StreamReader f = new StreamReader(path);

string record;

if (File.Exists(path))

{

while ((record = f.ReadLine()) != null)

{

string[] splittedRecord = record.Split(',');

string name = splittedRecord[0];

string password = splittedRecord[1];

string id = splittedRecord[2];

string userType = splittedRecord[3];

DateTime dob = DateTime.Parse(splittedRecord[4]);

string email = splittedRecord[5];

Int64 phone = Int64.Parse(splittedRecord[6]);

string[] splittedRecordForMedicine = splittedRecord[7].Split(';');

Customer customer = new Customer(name, password, id, userType,dob,email,phone);

for (int i = 0; i < splittedRecordForMedicine.Length; i++)

{

Medicine m = MedicineDL.searcher(splittedRecordForMedicine[i]);

if (m != null)

{

customer.setMedicine(m);

}

}

setList(customer);

}

f.Close();

return true;

}

else { return false; }

}

public static void storeIntoFile(string path, Customer customer)

{

string medicineName = string.Join(";", customer.getMedicine().Select(medicine => medicine.Name));

List<string> fileData = new List<string>();

if (File.Exists(path))

{

fileData = File.ReadAllLines(path).ToList();

for (int i = 0; i < fileData.Count; i++)

{

string[] customerInfo = fileData[i].Split(',');

if (customerInfo[0] == customer.Name)

{

List<string> existingMedicines = customerInfo[7].Split(';').ToList();

existingMedicines.AddRange(customer.getMedicine().Select(medicine => medicine.Name));

customerInfo[4] = string.Join(";", existingMedicines.Distinct());

fileData[i] = string.Join(",", customerInfo);

break;

}

}

}

if (!fileData.Any(line => line.StartsWith(customer.Name + ",")))

{

string customerInfo = $"{customer.Name},{customer.Password},{customer.Id},{customer.UserType},{customer.DOB},{customer.Email},{customer.PhoneNo},{medicineName}";

fileData.Add(customerInfo);

}

File.WriteAllLines(path, fileData);

}

}

}

**Forms Code**

**Pharmacist Forms:**

**Add Medicine Form:**

using Pharmacy\_Management\_System.BL;

using Pharmacy\_Management\_System.DL;

using System;

using System.Windows.Forms;

namespace Pharmacy\_Management\_System.PharmistUC

{

public partial class UC\_P\_AddMedicine : UserControl

{

public UC\_P\_AddMedicine()

{

InitializeComponent();

}

private void btnReset\_Click(object sender, EventArgs e)

{

clearAll();

}

private void clearAll()

{

pictureBox1.Visible = false;

txtExpiryDate.ResetText();

txtManufacturingDate.ResetText();

txtMedicineName.ResetText();

txtMedicineNumber.ResetText();

txtMedicinneID.ResetText();

txtPrice.ResetText();

txtQuantity.ResetText();

}

private void txtMedicinneID\_TextChanged(object sender, EventArgs e)

{

pictureBox1.Visible = true;

Medicine data = MedicineDL.searcher(txtMedicinneID.Text);

if (data == null)

{

pictureBox1.ImageLocation = @"D:\BS CS\PROJECTs SEMESTER2\OOP\GUNA File\Pharmacy Management System in C#\yes.png";

}

else

{

pictureBox1.ImageLocation = @"D:\BS CS\PROJECTs SEMESTER2\OOP\GUNA File\Pharmacy Management System in C#\no.png";

}

}

private void btnAdd\_Click(object sender, EventArgs e)

{

string medID = txtMedicinneID.Text;

string name = txtMedicineName.Text;

string medNum = txtMedicineNumber.Text;

DateTime manufactureDate = DateTime.Parse(txtManufacturingDate.Text);

DateTime expiryDate = DateTime.Parse(txtExpiryDate.Text);

int quantity = int.Parse(txtQuantity.Text);

int price = int.Parse(txtPrice.Text);

Medicine med = MedicineDL.searcher(medID);

if (med!=null)

{

MessageBox.Show("Sorry, Another Medicine With Same Id Already Exists !", "Fail", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

Medicine medicine = new Medicine(medID, name, medNum, manufactureDate, expiryDate, quantity, price);

MedicineDL.setList(medicine);

MedicineDL.storeintoFile(Program.medPath);

MessageBox.Show("Medicine added successfully !", "Succes", MessageBoxButtons.OK, MessageBoxIcon.Information);

clearAll();

}

}

private void txtQuantity\_KeyPress(object sender, KeyPressEventArgs e)

{

Program.integerValidation(e);

}

private void txtPrice\_KeyPress(object sender, KeyPressEventArgs e)

{

Program.integerValidation(e);

}

private void txtMedicineNumber\_KeyPress(object sender, KeyPressEventArgs e)

{

Program.integerValidation(e);

}

}

}

**Dashboard Form:**

using System;

namespace Pharmacy\_Management\_System.PharmistUC

{

public partial class UC\_P\_Dashboard : UserControl

{

public UC\_P\_Dashboard()

{

InitializeComponent();

}

private void UC\_P\_Dashboard\_Load(object sender, EventArgs e)

{

LoadChart();

}

public void LoadChart()

{

DateTime currentDate = DateTime.Now;

Int64 count1 = 0;

Int64 count2 = 0;

foreach (Medicine medicine in MedicineDL.getList())

{

if (medicine.Expiry\_date >= currentDate)

{

count1 ++;

}

}

this.chart1.Series["Valid Medicines"].Points.AddXY("Medicine Validity Chart", count1);

foreach (Medicine medicine in MedicineDL.getList())

{

if (medicine.Expiry\_date <= currentDate)

{

count2++;

}

}

this.chart1.Series["Expired Medicines"].Points.AddXY("Medicine Validity Chart", count2);

}

private void btnReload\_Click(object sender, EventArgs e)

{

chart1.Series["Valid Medicines"].Points.Clear();

chart1.Series["Expired Medicines"].Points.Clear();

LoadChart();

}

}

}

**Valid Medicines View Form:**

using System;

namespace Pharmacy\_Management\_System.PharmistUC

{

public partial class UC\_P\_MedicineValidityCheck : UserControl

{

public UC\_P\_MedicineValidityCheck()

{

InitializeComponent();

}

private void txtCheck\_SelectedIndexChanged(object sender, EventArgs e)

{

DateTime currentDate = DateTime.Now;

List<Medicine> list = new List<Medicine>();

if (txtCheck.SelectedIndex==0)

{

foreach (Medicine medicine in MedicineDL.getList())

{

if (medicine.Expiry\_date >= currentDate)

{

list.Add(medicine);

}

}

setDataGrid("Valid Medicines", Color.Black, list);

}

else if (txtCheck.SelectedIndex == 1)

{

foreach (Medicine medicine in MedicineDL.getList())

{

if (medicine.Expiry\_date <= currentDate)

{

list.Add(medicine);

}

}

setDataGrid("Expired Medicines", Color.Red, list);

}

else if (txtCheck.SelectedIndex == 2)

{

setDataGrid(null, Color.Black, MedicineDL.getList());

}

}

private void setDataGrid(string labelName,Color color,List<Medicine> list)

{

guna2DataGridView1.DataSource = list;

setLabel.Text = labelName;

setLabel.ForeColor = color;

}

private void UC\_P\_MedicineValidityCheck\_Load(object sender, EventArgs e)

{

setLabel.Text = null;

}

}

}

**Sell Medicines Form:**

using System;

namespace Pharmacy\_Management\_System.PharmistUC

{

public partial class UC\_P\_SellMedicine : UserControl

{

public UC\_P\_SellMedicine()

{

InitializeComponent();

guna2DataGridView1.Rows[n].Cells[0].ReadOnly = true;

guna2DataGridView1.Rows[n].Cells[1].ReadOnly = true;

guna2DataGridView1.Rows[n].Cells[2].ReadOnly = true;

guna2DataGridView1.Rows[n].Cells[3].ReadOnly = true;

guna2DataGridView1.Rows[n].Cells[4].ReadOnly = true;

guna2DataGridView1.Rows[n].Cells[5].ReadOnly = true;

}

private void UC\_P\_SellMedicine\_Load(object sender, EventArgs e)

{

ListBoxMedicines.Items.Clear();

DateTime currentDate = DateTime.Now;

foreach (Medicine item in MedicineDL.getList())

{

if (item.Expiry\_date >= currentDate && item.Quantity > 0)

{

ListBoxMedicines.Items.Add(item.Name.ToString());

}

}

}

private void btnSync\_Click(object sender, EventArgs e)

{

UC\_P\_SellMedicine\_Load(this, null);

}

private void txtSearch\_TextChanged(object sender, EventArgs e)

{

ListBoxMedicines.Items.Clear();

List<Medicine> medicines = MedicineDL.getList();

List<Medicine> matchingMedicines = new List<Medicine>();

foreach (Medicine medicine in medicines)

{

if (medicine.Name.StartsWith(txtSearch.Text) || medicine.Name.Contains(txtSearch.Text))

{

matchingMedicines.Add(medicine);

}

}

foreach (Medicine medicine in matchingMedicines)

{

ListBoxMedicines.Items.Add(medicine.Name.ToString());

}

}

private void ListBoxMedicines\_SelectedIndexChanged(object sender, EventArgs e)

{

txtUnits.Clear();

string name = ListBoxMedicines.GetItemText(ListBoxMedicines.SelectedItem);

txtMedName.Text = name;

Medicine medicine = MedicineDL.searchByName(name);

if (medicine != null)

{

txtMedID.Text = medicine.MedicineID.ToString();

txtExpiryDate.Text = medicine.Expiry\_date.ToString();

txtPrice.Text = medicine.Price.ToString();

}

else

{

MessageBox.Show("No Medicine Found.");

}

}

private void txtUnits\_TextChanged(object sender, EventArgs e)

{

int noOfUnits;

if (int.TryParse(txtUnits.Text, out noOfUnits))

{

int unitPrice = int.Parse(txtPrice.Text);

int Total = unitPrice \* noOfUnits;

txtTotal.Text = Total.ToString();

}

else

{

txtTotal.Clear();

}

}

protected int n, totalAmount = 0;

protected int quantity, newQuantity;

int valueAmount;

string valueID;

protected int noOfUnit;

private void guna2DataGridView1\_CellClick(object sender, DataGridViewCellEventArgs e)

{

try

{

valueAmount = int.Parse(guna2DataGridView1.Rows[e.RowIndex].Cells[6].Value.ToString());

valueID = guna2DataGridView1.Rows[e.RowIndex].Cells[0].Value.ToString();

noOfUnit = int.Parse(guna2DataGridView1.Rows[e.RowIndex].Cells[4].Value.ToString());

}

catch (Exception)

{

//throw;

}

}

private void btnRemove\_Click(object sender, EventArgs e)

{

if (valueID != null)

{

try

{

guna2DataGridView1.Rows.RemoveAt(this.guna2DataGridView1.SelectedRows[0].Index);

}

catch (Exception)

{

//throw;

}

finally

{

Medicine medicine = MedicineDL.searcher(valueID);

quantity = medicine.Quantity;

newQuantity = quantity + noOfUnit;

medicine.Quantity = newQuantity;

MedicineDL.storeintoFile(Program.medPath);

MessageBox.Show("Selected Medicine Removed from Cart."); totalAmount = totalAmount - valueAmount;

totalLabel.Text = "Rs. " + totalAmount.ToString();

}

UC\_P\_SellMedicine\_Load(this, null);

}

}

private void btnPurchase\_Click(object sender, EventArgs e)

{

MessageBox.Show("Selected Bill Printed.", "Information");

totalAmount = 0;

totalLabel.Text = "Rs. 00";

guna2DataGridView1.DataSource = 0;

}

private void txtPrice\_KeyPress(object sender, KeyPressEventArgs e)

{

Program.integerValidation(e);

}

private void txtUnits\_KeyPress(object sender, KeyPressEventArgs e)

{

Program.integerValidation(e);

}

private void txtTotal\_KeyPress(object sender, KeyPressEventArgs e)

{

Program.integerValidation(e);

}

private void btnSearch\_Click(object sender, EventArgs e)

{

ListBoxMedicines.Items.Clear();

string medicineName = txtSearch.Text;

foreach (Medicine medicine in MedicineDL.getList())

{

if (medicineName == medicine.Name)

{

ListBoxMedicines.Items.Add(medicineName.ToString());

}

}

}

private void btnAddToCart\_Click(object sender, EventArgs e)

{

if (txtMedID.Text != null)

{

string ID = txtMedID.Text;

Medicine medicine = MedicineDL.searcher(ID);

if (medicine == null)

{

MessageBox.Show("No Medicine Found.", "information");

}

else

{

quantity = medicine.Quantity;

int selectedQuantity = int.Parse(txtUnits.Text);

newQuantity = quantity - selectedQuantity;

if (newQuantity >= 0)

{

n = guna2DataGridView1.Rows.Add();

guna2DataGridView1.Rows[n].Cells[0].Value = txtMedID.Text;

guna2DataGridView1.Rows[n].Cells[1].Value = txtMedName.Text;

guna2DataGridView1.Rows[n].Cells[2].Value = txtExpiryDate.Text;

guna2DataGridView1.Rows[n].Cells[3].Value = txtPrice.Text;

guna2DataGridView1.Rows[n].Cells[4].Value = txtUnits.Text;

guna2DataGridView1.Rows[n].Cells[5].Value = txtTotal.Text;

totalAmount = totalAmount + int.Parse(txtTotal.Text);

totalLabel.Text = "Rs. " + totalAmount.ToString();

medicine.Quantity = newQuantity;

MedicineDL.storeintoFile(Program.medPath);

}

else

{

MessageBox.Show("Medicine is Out of Stock.\n Only " + quantity + " left", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

clearAll();

UC\_P\_SellMedicine\_Load(this, null);

}

}

else

{

MessageBox.Show("Select Medicine First.", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

private void clearAll()

{

txtMedID.Clear();

txtMedName.Clear();

txtExpiryDate.ResetText();

txtPrice.Clear();

txtUnits.Clear();

}

}

}

**Update Medicines Form:**

using System;

namespace Pharmacy\_Management\_System.PharmistUC

{

public partial class UC\_P\_UpdateMedicine : UserControl

{

public UC\_P\_UpdateMedicine()

{

InitializeComponent();

}

private void btnSearch\_Click(object sender, EventArgs e)

{

if (txtMedicineID.Text!=null)

{

string ID = txtMedicineID.Text;

Medicine medicine = MedicineDL.searcher(ID);

if (medicine != null)

{

txtMedicineName.Text = medicine.Name.ToString();

txtMedicineNumber.Text = medicine.MedicineNumber.ToString();

txtPrice.Text = medicine.Price.ToString();

txtManufacturingDate.Text = medicine.Manufacturing\_date.ToString();

txtExpiryDate.Text = medicine.Expiry\_date.ToString();

txtAvailableQuantity.Text = medicine.Quantity.ToString();

}

else

{

MessageBox.Show("No Medicine with ID : " + txtMedicineID.Text + "exists.", "Info", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

else

{

clearAll();

}

}

private void clearAll()

{

txtMedicineID.Clear();

txtMedicineName.Clear();

txtPrice.Clear();

txtManufacturingDate.ResetText();

txtExpiryDate.ResetText();

if (txtAddQuantity.Text!="0")

{

txtAddQuantity.Text = "0";

}

else

{

txtAddQuantity.Text = "0";

}

txtMedicineNumber.ResetText();

txtAvailableQuantity.ResetText();

}

private void btnReset\_Click(object sender, EventArgs e)

{

clearAll();

}

int totalQuantity;

private void btnUpdate\_Click(object sender, EventArgs e)

{

string name = txtMedicineName.Text;

Int64 price = Int64.Parse(txtPrice.Text);

string number = txtMedicineNumber.Text;

DateTime mDate = DateTime.Parse(txtManufacturingDate.Text);

DateTime expiryDate = DateTime.Parse(txtExpiryDate.Text);

int availableQuantity = int.Parse(txtAvailableQuantity.Text);

int addQuantity = int.Parse(txtAddQuantity.Text);

totalQuantity = availableQuantity + addQuantity;

string ID = txtMedicineID.Text;

Medicine medicine = MedicineDL.searcher(ID);

if (medicine != null)

{

medicine.Name = name;

medicine.Price = price;

medicine.MedicineNumber = number;

medicine.Manufacturing\_date = mDate;

medicine.Expiry\_date = expiryDate;

medicine.Quantity = totalQuantity;

MedicineDL.storeintoFile(Program.medPath);

MessageBox.Show("Medicine Updation Complete !");

}

else

{

MessageBox.Show("No Medicine with ID : " + txtMedicineID + "exists.", "Info", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

private void txtMedicineNumber\_KeyPress(object sender, KeyPressEventArgs e)

{

Program.integerValidation(e);

}

private void txtAvailableQuantity\_KeyPress(object sender, KeyPressEventArgs e)

{

Program.integerValidation(e);

}

private void txtPrice\_KeyPress(object sender, KeyPressEventArgs e)

{

Program.integerValidation(e);

}

private void txtAddQuantity\_KeyPress(object sender, KeyPressEventArgs e)

{

Program.integerValidation(e);

}

}

}

**View Medicines Form:**

using Pharmacy\_Management\_System.BL;

using Pharmacy\_Management\_System.DL;

using System;

using System.Collections.Generic;

namespace Pharmacy\_Management\_System.PharmistUC

{

public partial class UC\_P\_ViewMedicine : UserControl

{

public UC\_P\_ViewMedicine()

{

InitializeComponent();

}

private void UC\_P\_ViewMedicine\_Load(object sender, EventArgs e)

{

refreshGrid();

}

private void txtSearch\_TextChanged(object sender, EventArgs e)

{

// Get the list of medicines.

List<Medicine> medicines = MedicineDL.getList();

// Create a new list to store the matching medicines.

List<Medicine> matchingMedicines = new List<Medicine>();

// Iterate through the list of medicines and add any medicine that matches the text in the text box to the new list.

foreach (Medicine medicine in medicines)

{

if (medicine.Name.StartsWith(txtSearch.Text) || medicine.Name.Contains(txtSearch.Text))

{

matchingMedicines.Add(medicine);

}

}

// Set the data grid source to the new list of matching medicines.

guna2DataGridView2.DataSource = matchingMedicines;

}

string medicineID;

private void btnDelete\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Are you sure ?", "Delete Conformation !", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) == DialogResult.Yes)

{

Medicine medicine = guna2DataGridView2.CurrentRow.DataBoundItem as Medicine;

MedicineDL.getList().Remove(medicine);

MedicineDL.storeintoFile(Program.medPath);

MessageBox.Show("User Record Deleted. ", "Succes", MessageBoxButtons.OK, MessageBoxIcon.Information);

refreshGrid();

}

}

private void guna2DataGridView2\_CellClick(object sender, DataGridViewCellEventArgs e)

{

try

{

medicineID = guna2DataGridView2.Rows[e.RowIndex].Cells[1].Value.ToString();

}

catch (Exception)

{

//throw;

}

}

private void refreshGrid()

{

guna2DataGridView2.DataSource = null;

guna2DataGridView2.DataSource = MedicineDL.getList();

guna2DataGridView2.Refresh();

}

private void btnSync\_Click(object sender, EventArgs e)

{

refreshGrid();

}

private void btnSearch\_Click(object sender, EventArgs e)

{

string medicineName = txtSearch.Text;

List<Medicine> medicines = MedicineDL.listSearcher(medicineName);

guna2DataGridView2.DataSource = null;

guna2DataGridView2.DataSource = medicines;

}

private void btnReset\_Click(object sender, EventArgs e)

{

txtSearch.Clear();

guna2DataGridView2.DataSource = null;

UC\_P\_ViewMedicine\_Load(this, null);

}

}

}

**Admin Form:**

using System;

namespace Pharmacy\_Management\_System

{

public partial class Administrator : Form

{

string user = "";

public Administrator()

{

InitializeComponent();

}

private string ID

{

get { return user.ToString(); }

}

public Administrator(BL.User username)

{

InitializeComponent();

UserNamelabel.Text = username.Name;

user = username.Id;

uC\_View\_User1.ID = ID;

uC\_Profile1.ID = ID;

}

private void btnLogout\_Click(object sender, EventArgs e)

{

Form1 form = new Form1();

form.Show();

this.Hide();

}

private void Administrator\_Load(object sender, EventArgs e)

{

uC\_Dashboard1.Visible = false;

uC\_AddUser1.Visible = false;uC\_View\_User1.Visible = false;

uC\_Profile1.Visible = false;btnDashboard.PerformClick();

}

private void btnDashboard\_Click(object sender, EventArgs e)

{

uC\_Dashboard1.Visible = true;

uC\_Dashboard1.BringToFront();

}

private void btnAddUser\_Click(object sender, EventArgs e)

{

uC\_AddUser1.Visible = true;

uC\_AddUser1.BringToFront();

}

private void btnViewUser\_Click(object sender, EventArgs e)

{

uC\_View\_User1.Visible = true;

uC\_View\_User1.BringToFront();

}

private void btnProfile\_Click(object sender, EventArgs e)

{

uC\_Profile1.Visible = true;

uC\_Profile1.BringToFront();

}

}

}

**Admin Forms:**

**Dashboard Form:**

using Microsoft.VisualBasic.ApplicationServices;

using Pharmacy\_Management\_System.DL;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Windows.Forms;

namespace Pharmacy\_Management\_System.AdministrartorUC

{

public partial class UC\_Dashboard : UserControl

{

public UC\_Dashboard()

{

InitializeComponent();

}

private void guna2Button1\_Click\_1(object sender, EventArgs e)

{

UC\_Dashboard\_Load(this, null);

}

private void UC\_Dashboard\_Load(object sender, EventArgs e)

{

AdminLabel.Text = getUsersCount(UserDL.getUsers(), "Administrator").ToString();

PharmistLabel.Text = getUsersCount(UserDL.getUsers(), "Pharmist").ToString();

CustomerLabel.Text = getUsersCount(UserDL.getUsers(), "user").ToString();

}

private int getUsersCount(List<BL.User> users, string role)

{

int count = 0;

foreach (BL.User user in UserDL.getUsers())

{

if (user.UserType == role)

{

count = count + 1;

}

}

return count;

}

}

}

**Add User Form:**

using Microsoft.VisualBasic.ApplicationServices;

using Pharmacy\_Management\_System.DL;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Pharmacy\_Management\_System.AdministrartorUC

{

public partial class UC\_AddUser : UserControl

{

private DataGridView userGrid;

public UC\_AddUser()

{

InitializeComponent();

}

private void btnReset\_Click(object sender, EventArgs e)

{

clearAll();

}

private void clearAll()

{

txtDOB.ResetText();

txtEmail.ResetText();

txtPassword.ResetText();

txtUserName.ResetText();

txtMobileNO.ResetText();

txtName.ResetText();

pictureBox1.Visible = false;

txtUserRole.SelectedIndex = -1;

}

private void txtUserName\_TextChanged\_1(object sender, EventArgs e)

{

pictureBox1.Visible = true;

BL.User data = UserDL.check(txtUserName.Text);

if (data != null)

{

pictureBox1.ImageLocation = @"D:\BS CS\PROJECTs SEMESTER2\OOP\GUNA File\Pharmacy Management System in C#\no.png";

}

else

{

pictureBox1.ImageLocation = @"D:\BS CS\PROJECTs SEMESTER2\OOP\GUNA File\Pharmacy Management System in C#\yes.png";

}

}

private void btnSignUp\_Click(object sender, EventArgs e)

{

string role = txtUserRole.Text;

string name = txtName.Text;

DateTime DOB = DateTime.Parse(txtDOB.Text);

long number = long.Parse(txtMobileNO.Text);

string email = txtEmail.Text;

string password = txtPassword.Text;

string userID = txtUserName.Text;

BL.User check = UserDL.check(userID);

if (check!=null)

{

MessageBox.Show("Sorry another user already exists with that id !", "Fail", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

BL.User user = new BL.User(role, name, DOB, number, email, password, userID);

UserDL.setUser(user);

UserDL.storeintoFile(Program.userPath);

MessageBox.Show("Signed Up successfully !", "Succes", MessageBoxButtons.OK, MessageBoxIcon.Information);

clearAll();

}

}

private void txtMobileNO\_KeyPress(object sender, KeyPressEventArgs e)

{

Program.integerValidation(e);

}

}

}

**View User Form:**

using Pharmacy\_Management\_System.BL;

using Pharmacy\_Management\_System.DL;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Pharmacy\_Management\_System.AdministrartorUC

{

public partial class UC\_View\_User : UserControl

{

string currentUser = "";

public UC\_View\_User()

{

InitializeComponent();

}

public string ID

{

set { currentUser = value; }

}

private void btnSync\_Click(object sender, EventArgs e)

{

refreshGrid();

}

private void UC\_View\_User\_Load(object sender, EventArgs e)

{

refreshGrid();

}

private void txtUserName\_TextChanged(object sender, EventArgs e)

{ List<User> users = UserDL.getUsers();

List<User> matchingUsers = new List<User>();

foreach (User user in users)

{

if (user.Name.StartsWith(txtUserName.Text) || user.Name.Contains(txtUserName.Text))

{

matchingUsers.Add(user);

}

}

guna2DataGridView1.DataSource = matchingUsers;

}

string usernameOrID;

private void guna2DataGridView1\_CellClick(object sender, DataGridViewCellEventArgs e)

{

try

{

usernameOrID = guna2DataGridView1.Rows[e.RowIndex].Cells[2].Value.ToString();

}

catch

{

//throw;

}

}

private void btnDelete\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Are you sure ?", "Delete Conformation !", MessageBoxButtons.YesNo, MessageBoxIcon.Warning) == DialogResult.Yes)

{

if (currentUser != usernameOrID)

{

User user = guna2DataGridView1.CurrentRow.DataBoundItem as User;

UserDL.getUsers().Remove(user);

UserDL.storeintoFile(Program.userPath);

MessageBox.Show("User Record Deleted. ", "Succes", MessageBoxButtons.OK, MessageBoxIcon.Information);

refreshGrid();

}

else

{

MessageBox.Show("You are trying to Delete \n Your own Profile.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

}

private void guna2DataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void refreshGrid()

{

guna2DataGridView1.DataSource = null;

guna2DataGridView1.DataSource = UserDL.getUsers();

guna2DataGridView1.Refresh();

}

private void btnSearch\_Click(object sender, EventArgs e)

{

string username = txtUserName.Text;

List<User> users = UserDL.searchUser(username);

guna2DataGridView1.DataSource = null;

guna2DataGridView1.DataSource = users;

}

private void btnReset\_Click(object sender, EventArgs e)

{

txtUserName.Clear();

guna2DataGridView1.DataSource = null;

UC\_View\_User\_Load(this, null);

}

}

}

**View Profile Form:**

using System;

namespace Pharmacy\_Management\_System.AdministrartorUC

{

public partial class UC\_Profile : UserControl

{

public UC\_Profile()

{

InitializeComponent();

}

public string ID

{

set { userNameLabel.Text = value; }

}

private void btnUpdate\_Click(object sender, EventArgs e)

{

string role = txtUserRole.Text;

string name = txtName.Text;

string password = txtPassword.Text;

string email = txtEmail.Text;

DateTime DOB = DateTime.Parse(txtDOB.Text);

long phone = int.Parse(txtPhone.Text);

BL.User user = UserDL.check(userNameLabel.Text);

user.UserType = role;

user.Email = email;

user.DOB = DOB;

user.Name = name;

user.Password = password;

user.PhoneNo = phone;

UserDL.storeintoFile(Program.userPath);

MessageBox.Show("Profile Updation Complete !", "Success", MessageBoxButtons.OK, MessageBoxIcon.Exclamation );

}

private void UC\_Profile\_Enter(object sender, EventArgs e)

{

BL.User user = UserDL.check(userNameLabel.Text);

txtUserRole.Text = user.UserType.ToString();

txtName.Text = user.Name.ToString();

txtEmail.Text = user.Email.ToString();

txtDOB.Text = user.DOB.ToString();

txtPassword.Text = user.Password.ToString();

txtPhone.Text = user.PhoneNo.ToString();

}

private void btnReset\_Click(object sender, EventArgs e)

{

UC\_Profile\_Enter(this, null);

}

private void txtPhone\_KeyPress(object sender, KeyPressEventArgs e)

{

Program.integerValidation(e);

}

}

}

**Pharmacist Form:**

using System;

namespace Pharmacy\_Management\_System

{

public partial class Pharmist : Form

{

public Pharmist()

{

InitializeComponent();

}

private void btnLogout\_Click(object sender, EventArgs e)

{

Form1 form = new Form1();

form.Show();

this.Hide();

}

private void btnDashboard\_Click(object sender, EventArgs e)

{

uC\_P\_Dashboard1.Visible = true;

uC\_P\_Dashboard1.BringToFront();

}

private void Pharmist\_Load(object sender, EventArgs e)

{

uC\_P\_Dashboard1.Visible = false;

uC\_P\_AddMedicine1.Visible = false;

uC\_P\_ViewMedicine1.Visible =false;uC\_P\_UpdateMedicine1.Visible = false;

uC\_P\_MedicineValidityCheck1.Visible = false;

uC\_P\_SellMedicine1.Visible=false;btnDashboard.PerformClick();

}

private void btnAddMedicine\_Click(object sender, EventArgs e)

{

uC\_P\_AddMedicine1.Visible = true;

uC\_P\_AddMedicine1.BringToFront();

}

private void btnViewMedicine\_Click(object sender, EventArgs e)

{

uC\_P\_ViewMedicine1.Visible = true;

uC\_P\_ViewMedicine1.BringToFront();

}

private void btnModifyMedicine\_Click(object sender, EventArgs e)

{

uC\_P\_UpdateMedicine1.Visible = true;

uC\_P\_UpdateMedicine1.BringToFront();

}

private void btnValidityCheck\_Click(object sender, EventArgs e)

{

uC\_P\_MedicineValidityCheck1.Visible = true;

uC\_P\_MedicineValidityCheck1.BringToFront();

}

private void btnSellMedicine\_Click(object sender, EventArgs e)

{

uC\_P\_SellMedicine1.Visible = true;

uC\_P\_SellMedicine1.BringToFront();

}

}

}

**Main Form:**

using Pharmacy\_Management\_System.BL;

using Pharmacy\_Management\_System.DL;

using System;

namespace Pharmacy\_Management\_System

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void btnExit\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void btnReset\_Click(object sender, EventArgs e)

{

clearAll();

}

private void clearAll()

{

txtUsername.Clear();

txtPassword.Clear();

}

private void btnSignIn\_Click\_1(object sender, EventArgs e)

{

User user = UserDL.login(txtUsername.Text, txtPassword.Text);

if (user != null)

{

if (user.UserType == "Administrator")

{

Administrator admin = new Administrator(user);

admin.Show();

this.Hide();

}

else if (user.UserType == "Pharmist")

{

Pharmist pharmist = new Pharmist();

pharmist.Show();

this.Hide();

}

}

if (user == null)

{

MessageBox.Show("Wrong Username or Password", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

clearAll();

}

if (UserDL.getUsers().Count == 0)

{

if (txtUsername.Text == "root" && txtPassword.Text == "root")

{

DialogResult result = MessageBox.Show("Do You want to Login as an Admin ? ", "Conformation", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (result == DialogResult.Yes)

{

Administrator admin = new Administrator();

admin.Show();

this.Hide();

}

else if (result == DialogResult.No)

{

Pharmist pharmist = new Pharmist();

pharmist.Show();

this.Hide();

}

}

else

{

MessageBox.Show("Wrong Username or Password", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

clearAll();

}

}

}

}

}

END