Hospital Management System

Session: 2022 – 2025

Submitted by:

Mudasir Ahmad 2022-CS-77

Supervised by:

Prof.Muhammad Awais Hassan

Prof. Laeeq Khan Niazi & Ma’am Maida Shahid

Department of Computer Science

University of Engineering and Technology

Lahore, Pakistan

Contents

YouTube Link:

https://youtu.be/r2KUGqOuCe8

Project Details:

Hospital Management System is used for managing hospital work because manual management is very difficult. It saves the time of patients and staff because the patient cannot need to stand in line. Hospital Management System is basically about adding Patients which is used by Management staff to perform different operations on their data such as generating reports, suggesting medicines, and calculating their bills. Management Staff performs the Patient **CRUD** operations. Management staff also manages the doctor's schedule and appointments. A patient can check a doctor prescription, view his medical report, and bill. If the patient has a common disease, this software asks for the symptoms and suggests the medicine according to the symptoms. This system also manages the Pharmacy store of the Hospital in which pharmacists can add their stock to the system, check the expired medicines, and view the medicines in demand. This system helps the hospital to run more efficiently by automating patient information and medical records, reducing errors, and improving financial management.

Users:

- The **management staff/Admin** of the hospital can use this application. This application can help them in reducing their work and provide more accuracy.
- The **pharmacist** can use it to manage stocks of medicine.
- **Patients** can use our application to see their medical report and expenses.

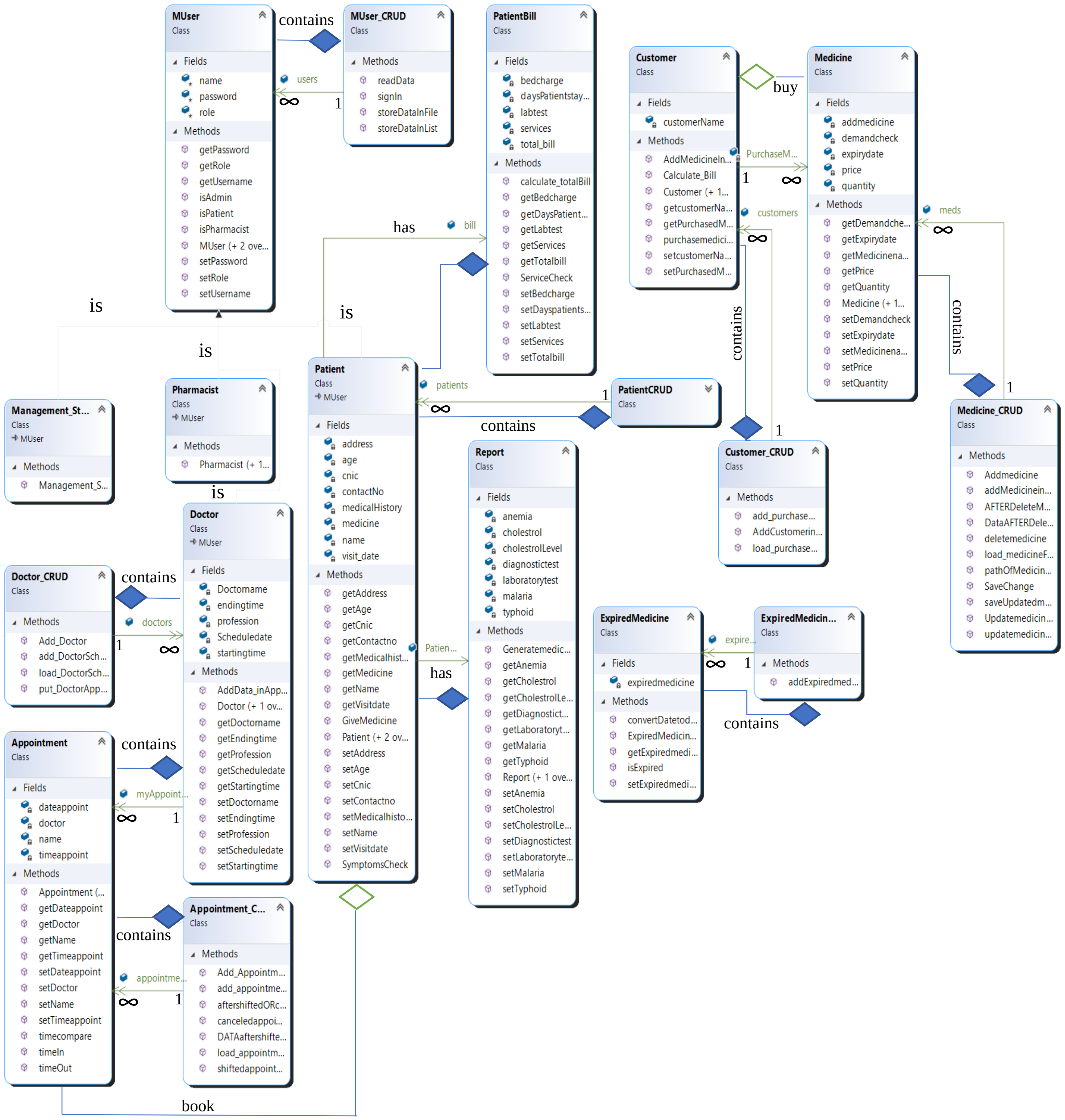
Functional Requirements:

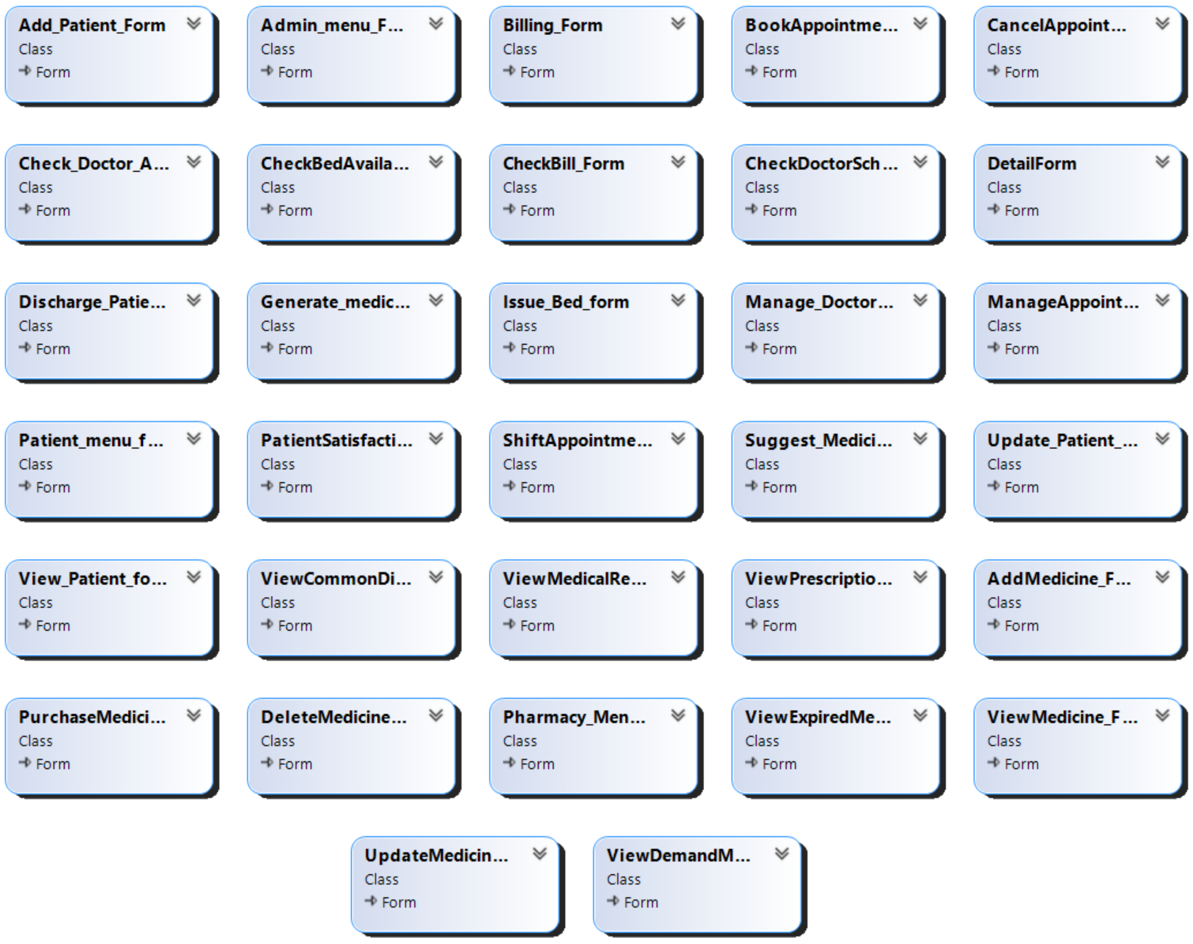
As a	***I want to perform***	***So that I can***
Management Staff	Add new patient Manage the doctor's schedule Manage the appointment with a doctor Add the record of expenses	Maintain his record Inform patients Save the time of patients Calculate the bill according to the services provided to a patient
Patient	View the medical report Check the bill	remain up-to-date about my health pay for it
Pharmacist	Check the medicines in demand Check the Expired medicines	Send a request to the supplier for those medicines Delete it from my record

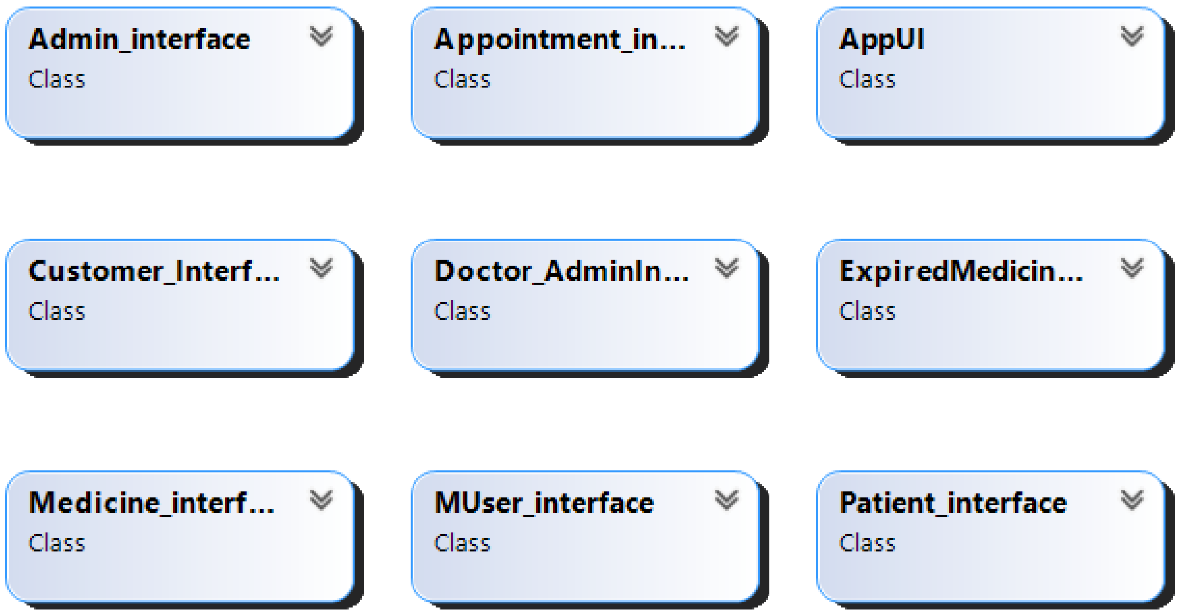
Project fulfilling OOP Concepts till Week 8:

I have used the Concepts of OOP in my project. Inheritance is a powerful feature that enables us to create hierarchical relationships between classes. I have used inheritance based on the different roles of users and used static polymorphism on validations. I have used the association such as aggregation and composition in different stages of this project. When the object of the patient is created then it automatically creates the object of its report and bill because there is no existence of a patient report without the patient this is composition. A patient can book many appointments with different doctors. A customer can purchase many medicines from a pharmacy store because these things are independent this is aggregation. OOP promotes modularity by breaking down complex problems into smaller, manageable components called objects. These objects encapsulate data and behavior, making it easier to understand, maintain, and modify code. Objects can also be reused in different parts of a program, leading to code reusability, which saves development time and effort.

CRC Diagram:


MUser
Class
Fields
name
password
role
Methods
getPassword
getRole
getUsername
isAdmin
isPatient
isPharmacist
MUser (+ 2 ove...
setPassword
setRole
setUsername
contains
users
∞
1
MUser_CRUD
Class
Methods
readData
signIn
storeDataInFile
storeDataInList
PatientBill
Class
Fields
bedcharge
daysPatientstay...
labtest
services
total_bill
Methods
calculate_totalBill
getBedcharge
getDaysPatient...
getLabtest
getServices
getTotalbill
ServiceCheck
setBedcharge
setDayspatients...
setLabtest
setServices
setTotalbill
has
bill
Customer
Class
Fields
customerName
Methods
AddMedicineIn...
Calculate_Bill
Customer (+ 1...
getcustomerNa...
getPurchasedM...
purchasemedici...
setcustomerNa...
setPurchasedM...
buy
PurchaseM...
1
∞
customers
∞
Medicine
Class
Fields
addmedicine
demandcheck
expirydate
price
quantity
Methods
getDemandche...
getExpirydate
getMedicinena...
getPrice
getQuantity
Medicine (+ 1...
setDemandcheck
setExpirydate
setMedicinena...
setPrice
setQuantity
meds
∞
contains
1
contains
is
is
is
is
Management_St...
Class
MUser
Methods
Management_S...
Pharmacist
Class
MUser
Methods
Pharmacist (+ 1...
Patient
Class
MUser
Fields
address
age
cnic
contactNo
medicalHistory
medicine
name
visit_date
Methods
getAddress
getAge
getCnic
getContactno
getMedicalhist...
getMedicine
getName
getVisitdate
GiveMedicine
Patient (+ 2 ov...
setAddress
setAge
setCnic
setContactno
setMedicalhisto...
setName
setVisitdate
SymptomsCheck
patients
∞
1
contains
PatientCRUD
Class
Report
Class
Fields
anemia
cholestrol
cholestrolLevel
diagnostictest
laboratorytest
malaria
typhoid
Methods
Generatemedic...
getAnemia
getCholestrol
getCholestrolLe...
getDiagnostict...
getLaboratoryt...
getMalaria
getTyphoid
Report (+ 1 ove...
setAnemia
setCholestrol
setCholestrolLe...
setDiagnostictest
setLaboratoryte...
setMalaria
setTyphoid
Patien...
has
Customer_CRUD
Class
Methods
add_purchase...
AddCustomerin...
load_purchase...
1
Medicine_CRUD
Class
Methods
Addmedicine
addMedicinein...
AFTERDeleteM...
DataAFTERDele...
deletemedicine
load_medicineF...
pathOfMedicin...
SaveChange
saveUpdatedm...
Updatemedicin...
updatemedicin...
Doctor
Class
MUser
Fields
Doctorname
endingtime
profession
Scheduledate
startingtime
Methods
AddData_inApp...
Doctor (+ 1 ov...
getDoctorname
getEndingtime
getProfession
getScheduledate
getStartingtime
setDoctorname
setEndingtime
setProfession
setScheduledate
setStartingtime
Doctor_CRUD
Class
Methods
Add_Doctor
add_DoctorSch...
load_DoctorSch...
put_DoctorApp...
contains
doctors
1
∞
ExpiredMedicine
Class
Fields
expiredmedicine
Methods
convertDatetod...
ExpiredMedicin...
getExpiredmedi...
isExpired
setExpiredmedi...
ExpiredMedicin...
Class
Methods
addExpiredmed...
expire...
∞
1
contains
Appointment
Class
Fields
dateappoint
doctor
name
timeappoint
Methods
Appointment (...
getDateappoint
getDoctor
getName
getTimeappoint
setDateappoint
setDoctor
setName
setTimeappoint
timecompare
timeIn
timeOut
contains
myAppoint...
∞
1
Appointment_C...
Class
Methods
Add_Appointm...
add_appointme...
aftershiftedORc...
canceledappoi...
DATAaftershifte...
load_appointm...
shiftedappoint...
contains
appointme...
∞
1
book


Add_Patient_Form
Class
Form
Admin_menu_F...
Class
Form
Billing_Form
Class
Form
BookAppointme...
Class
Form
CancelAppoint...
Class
Form
Check_Doctor_A...
Class
Form
CheckBedAvaila...
Class
Form
CheckBill_Form
Class
Form
CheckDoctorSch...
Class
Form
DetailForm
Class
Form
Discharge_Patie...
Class
Form
Generate_medic...
Class
Form
Issue_Bed_form
Class
Form
Manage_Doctor...
Class
Form
ManageAppoint...
Class
Form
Patient_menu_f...
Class
Form
PatientSatisfacti...
Class
Form
ShiftAppointme...
Class
Form
Suggest_Medici...
Class
Form
Update_Patient_...
Class
Form
View_Patient_fo...
Class
Form
ViewCommonDi...
Class
Form
ViewMedicalRe...
Class
Form
ViewPrescriptio...
Class
Form
AddMedicine_F...
Class
Form
PurchaseMedici...
Class
Form
DeleteMedicine...
Class
Form
Pharmacy_Men...
Class
Form
ViewExpiredMe...
Class
Form
ViewMedicine_F...
Class
Form
UpdateMedicin...
Class
Form
ViewDemandM...
Class
Form



Code:

BL Code: MUser.cs

```
class MUser
    {
        protected string name;
        protected string password;
        protected string role;

        public MUser()
        {
            name = "";
            password = "";
            role = "";
        }
        public MUser(string name, string password)
        {
            this.name = name;
            this.password = password;
        }
        public MUser(string name, string password, string role)
        {
            this.name = name;
            this.password = password;
            this.role = role;
        }
        public string getUsername()
        {
            return name;
        }
          public string getPassword()
        {
            return password;
        }
        public string getRole()
        {
            return role;
        }
        public void setUsername(string name)
        {
            this.name = name;
        }
        public void setPassword(string password)
        {
            this.password = password;
        }
        public void setRole(string role)
        {
            this.role = role;
        }
        public bool isAdmin()
        {
            if (role == "Admin" || role == "admin")
            {
                return true;
            }
            return false;
        }
        public bool isPatient()
        {
            if (role == "Patient" || role == "patient")
            {
                return true;
            }
            return false;
        }
        public bool isPharmacist()
        {
            if (role == "Pharmacist" || role == "pharmacist")
            {
                return true;
            }
            return false;
        }
    }
```

BL Code: Patient.cs

```
class Patient : MUser
    {

        public Patient(string username, string password, string role, string name,
int age, string contactNo, string cnic, string address, string visit_date, string
medicalHistory) : base(username, password, role)
        {
            this.name = name;
            this.age = age;
            this.contactNo = contactNo;
            this.cnic = cnic;
            this.address = address;
            this.visit_date = visit_date;
            this.medicalHistory = medicalHistory;
        }
        public Patient(string username, string password, string role) :
base(username, password, role)
        {

        }
        public Patient()
        {

        }
        private string name;
        private int age;
        private string contactNo;
        private string cnic;
        private string address;
        private string visit_date;
        private string medicalHistory;
        public Report PatientReport = new Report();
        public PatientBill bill = new PatientBill();

        public string getName() { return name; }
        public int getAge() { return age; }
        public string getContactno() { return contactNo; }
        public string getCnic() { return cnic; }
        public string getAddress() { return address; }
        public string getVisitdate() { return visit_date; }
        public string getMedicalhistory() { return medicalHistory; }

        public void setName(string name) { this.name = name; }
        public void setAge(int age) { this.age = age; }
        public void setContactno(string contactNo) { this.contactNo = contactNo; }
        public void setCnic(string cnic) { this.cnic = cnic; }
        public void setAddress(string address) { this.address = address; }
        public void setVisitdate(string visit_date) { this.visit_date =
visit_date; }
        public void setMedicalhistory(string medicalHistory) { this.medicalHistory
= medicalHistory; }



        private string medicine;
        public string getMedicine() { return medicine; }
        public  void SymptomsCheck(string[] symptoms,int i,ref char check1,ref
char check2,ref char check3)
        {
            if (symptoms[i] == "fever" || symptoms[i] == "pain" || symptoms[i] ==
"headache")
            {
                check1 = 'p';
            }
            if (symptoms[i] == "cold" || symptoms[i] == "flu")
            {
                check2 = 'c';
            }
            if (symptoms[i] == "cough")
            {
                check3 = 'h';
            }
        }
        public string GiveMedicine(char check1,char check2,char check3)
        {
            string medicine1 = "",medicine2="",medicine3="";
            if (check1 == 'p')
            {
                medicine1 = " Panadol";
            }
            else
            {
                medicine1 = "";
            }
            if (check2 == 'c')
            {
                medicine2 = " Crocin C&F max";
            }
            else
            {
                medicine2 = "";
            }
            if (check3 == 'h')
            {
                medicine3 = " Hydryllin";
            }
            else
            {
                medicine3 = "";
            }
           return medicine = medicine1 + medicine2 + medicine3;

        }

    }
```

BL Code: Management staff.cs

```
class Management_Staff:MUser
    {
        public Management_Staff(string username, string password, string role) :
base(username, password, role)
        {

        }
        public Management_Staff(string username, string password) : base(username,
password)
        {

        }

    }
```

BL Code: Pharmacist.cs

```
class Pharmacist:MUser
    {
        public Pharmacist(string username, string password, string role) :
base(username, password, role)
        {

        }
        public Pharmacist(string username, string password) : base(username,
password)
        {

        }
    }
```

BL Code: Doctor.cs

```
class Doctor:MUser
    {
        public Doctor(string username, string password, string role,string
Scheduledate,string Doctorname,string profession,string startingtime,string
endingtime):base(username,password,role)
        {
            this.Scheduledate = Scheduledate;
            this.Doctorname = Doctorname;
            this.profession = profession;
            this.startingtime = startingtime;
            this.endingtime = endingtime;

        }
        public Doctor() { }
        private string Scheduledate, Doctorname, profession, startingtime,
endingtime;
        public List<Appointment> myAppointments = new List<Appointment>();
        public string getScheduledate() { return Scheduledate; }
        public string getDoctorname() { return Doctorname; }
        public string getProfession() { return profession; }
        public string getStartingtime() { return startingtime; }
        public string getEndingtime() { return endingtime; }
        public void setScheduledate(string Scheduledate) { this.Scheduledate =
Scheduledate; }
        public void setDoctorname(string Doctorname) { this.Doctorname =
Doctorname; }
        public void setProfession(string profession) { this.profession =
profession; }
        public void setStartingtime(string startingtime) { this.startingtime =
startingtime; }
        public void setEndingtime(string endingtime) { this.endingtime =
endingtime; }

        public void AddData_inAppointmentList(List<Appointment> appointments)
        {
            bool check = false;
            foreach(var appointment in appointments)
            {
foreach (var Appoinment in myAppointments)
{
 if (Appoinment.getName() != appointment.getName() ||       Appoinment.getDoctor()
!= appointment.getDoctor())
{
                        check = false;
                    }
                    else
                    {
                        check = true;
                        break;
                    }

                }
                if (check == false)
                {
                    if (Doctorname == appointment.getDoctor())
                    {
                        myAppointments.Add(appointment);
                    }
                }
            }
        }
    }
```

BL Code: Appointment.cs

```
class Appointment
    {
        private string name;
        private string dateappoint;
        private string doctor;
        private string timeappoint;
        public string getName() { return name; }
        public string getDateappoint() { return dateappoint; }
        public string getDoctor() { return doctor; }
        public string getTimeappoint() { return timeappoint; }
        public void setName(string name) { this.name = name; }
        public void setDateappoint(string dateappoint) { this.dateappoint =
dateappoint; }
        public void setDoctor(string doctor) { this.doctor = doctor; }
        public void setTimeappoint(string timeappoint) { this.timeappoint =
timeappoint; }
        public Appointment() { }
        public Appointment(string name, string dateappoint, string doctor)
        {
            this.name = name;
            this.dateappoint = dateappoint;
            this.doctor = doctor;
        }
        public Appointment(string name, string dateappoint, string doctor,string
timeappoint)
        {
            this.name = name;
            this.dateappoint = dateappoint;
            this.doctor = doctor;
            this.timeappoint = timeappoint;
        }

        public int Converttime(string timeappoint)
        {
            int time = 0;
            if (timeappoint[0] == '0')
            {
                time = 0;
            }
            if (timeappoint[0] == '1')
            {
                time = 1;
            }
            if (timeappoint[0] == '2')
            {
                time = 2;
            }
            if (timeappoint[0] == '3')
            {
                time = 3;
            }
            if (timeappoint[0] == '4')
            {
                time = 4;
            }
            if (timeappoint[0] == '5')
            {
                time = 5;
            }
            if (timeappoint[0] == '6')
            {
                time = 6;
            }
            if (timeappoint[0] == '7')
            {
                time = 7;
            }
            if (timeappoint[0] == '8')
            {
                time = 8;
            }
            if (timeappoint[0] == '9')
            {
                time = 9;
            }
            if (timeappoint[0] == '1' && timeappoint[1] == '0')
            {
                time = 10;
            }
            if (timeappoint[0] == '1' && timeappoint[1] == '1')
            {
                time = 11;
            }
            if (timeappoint[0] == '1' && timeappoint[1] == '2')
            {
                time = 12;
            }
            return time;
        }
```

BL Code: PatientBill.cs

```
class PatientBill
    {
        //private string PatientnameBill;
        private string services;
        private int daysPatientstayed;
        private float bedcharge;
        private float labtest;
        private float total_bill;
        public string getServices() { return services; }
        public int getDaysPatientstayed() { return daysPatientstayed; }
        public float getBedcharge() { return bedcharge; }
        public float getLabtest() { return labtest; }
        public float getTotalbill() { return total_bill; }

        public void setServices(string services) { this.services = services; }
        public void setDayspatientstayed(int daysPatientstayed) {
this.daysPatientstayed = daysPatientstayed; }
        public void setBedcharge(float bedcharge) { this.bedcharge = bedcharge; }
        public void setLabtest(float labtest) { this.labtest = labtest; }
        public void setTotalbill(float total_bill) { this.total_bill = total_bill;
}
        public int ServiceCheck()
        {
            int hospitalizationcharge = 0;
            int emergency_room_charge = 0;
            int final_charges;

            if (services == "hospitalization")
            {
                hospitalizationcharge = 500;
            }
            if (services == "emergency room service")
```

```
                {
                    emergency_room_charge = 2000;
                }
                return final_charges = hospitalizationcharge + emergency_room_charge;
            }
            public void calculate_totalBill(int hospitalizationcharge,int
emergency_room_charge,float bedcharge,float totalpricemedicine)
            {
                total_bill = (hospitalizationcharge * daysPatientstayed) +
(emergency_room_charge * daysPatientstayed) + (bedcharge * daysPatientstayed) +
(labtest + totalpricemedicine);
            }
        }
```

BL Code: Report.cs

```
class Report
    {
        private string laboratorytest;
        private int cholestrolLevel;
        private string cholestrol;
        private string typhoid;
        private string malaria;
        private string anemia;
        private string diagnostictest;

        public Report() { }
        public Report(string laboratorytest, int cholestrolLevel,string
cholestrol,string typhoid,string malaria,string anemia,string diagnostictest)
        {
            this.laboratorytest = laboratorytest;
            this.cholestrolLevel = cholestrolLevel;
            this.cholestrol = cholestrol;
            this.typhoid = typhoid;
            this.malaria = malaria;
            this.anemia = anemia;
            this.diagnostictest = diagnostictest;
        }
        public string getLaboratorytest() { return laboratorytest; }
        public int getCholestrolLevel() { return cholestrolLevel; }
        public string getCholestrol() { return cholestrol; }
        public string getTyphoid() { return typhoid; }
        public string getMalaria() { return malaria; }
        public string getAnemia() { return anemia; }
        public string getDiagnostictest() { return diagnostictest; }

        public void setLaboratorytest(string laboratorytest) { this.laboratorytest
= laboratorytest; }
        public void setCholestrolLevel(int cholestrolLevel) { this.cholestrolLevel
= cholestrolLevel; }
        public void setCholestrol(string cholestrol) { this.cholestrol =
cholestrol; }
        public void setTyphoid(string typhoid) { this.typhoid = typhoid; }
        public void setMalaria(string malaria) { this.malaria = malaria; }
        public void setAnemia(string anemia) { this.anemia = anemia; }
        public void setDiagnostictest(string diagnostictest) { this.diagnostictest
= diagnostictest; }

        public void Generatemedicalreport()
        {

            if (laboratorytest == "blood test")
            {
                if (cholestrolLevel < 100)
                {
                    cholestrol = "Optimum";
                }
                if (cholestrolLevel >= 100 && cholestrolLevel < 130)
                {
                    cholestrol = "Fairly Good";
                }
                if (cholestrolLevel >= 130 && cholestrolLevel < 160)
                {
                    cholestrol = "Borederline High";
                }
                if (cholestrolLevel >= 160 && cholestrolLevel < 190)
                {
                    cholestrol = "High";
                }
                if (cholestrolLevel >= 190)
                {
                    cholestrol = "Very High";
                }
            }
        }
    }
```

BL Code: Medicine.cs

```
class Medicine
    {
        private string addmedicine;
        private int quantity;
        private string expirydate;
        private float price;
        private int demandcheck;
        public string getMedicinename() { return addmedicine; }
        public int getQuantity() { return quantity; }
        public string getExpirydate() { return expirydate; }
        public float getPrice() { return price; }
        public int getDemandcheck() { return demandcheck; }
        public void setMedicinename(string addmedicine) { this.addmedicine =
addmedicine; }
        public void setQuantity(int quantity) { this.quantity = quantity; }
        public void setExpirydate(string expirydate) { this.expirydate =
expirydate; }
        public void setPrice(float price) { this.price = price; }
        public void setDemandcheck(int demandcheck) { this.demandcheck =
demandcheck; }
        public Medicine()
        {
            addmedicine = "";
            quantity = 0;
            expirydate = "";
            price = 0.0F;
            demandcheck = 0;
        }
        public Medicine(string addmedicine, int quantity, string expirydate, float
price, int demandcheck)
        {
            this.addmedicine = addmedicine;
            this.quantity = quantity;
            this.expirydate = expirydate;
            this.price = price;
            this.demandcheck = demandcheck;
        }


    }
```

BL Code: Customer.cs

```
class Customer
    {
       private string customerName;
       private List<Medicine> PurchaseMedicines = new List<Medicine>();
        public Customer(string customerName)
        {
            this.customerName = customerName;
        }
        public Customer()
        {

        }
        public string getcustomerName()
        {
            return customerName;
        }
        public void setcustomerName(string customerName)
        {
            this.customerName = customerName;
        }
        public List<Medicine> getPurchasedMedicines()
        {
            return PurchaseMedicines;
        }
        public void setPurchasedMedicines(List<Medicine> PurchaseMedicines)
        {
            this.PurchaseMedicines = PurchaseMedicines;
        }
        public void AddMedicineInList(Medicine PurchaseMed)
        {
            PurchaseMedicines.Add(PurchaseMed);
        }
        public Medicine purchasemedicine(List<Medicine> med, Medicine
purchasemed1)
        {
            foreach (Medicine i in med)
            {
                if (purchasemed1.getMedicinename() == i.getMedicinename())
                {
                    if (purchasemed1.getQuantity() <= i.getQuantity())
                    {
                        int final_quantity = i.getQuantity() - 
purchasemed1.getQuantity();
                        i.setQuantity(final_quantity);
                        purchasemed1.setExpirydate(i.getExpirydate());
                        purchasemed1.setPrice(i.getPrice());
                        purchasemed1.setDemandcheck(i.getDemandcheck());
                        return i;
                    }
                }
            }
            return null;
        }
        public float Calculate_Bill()
        {
            float total_price = 0F;
            foreach(var med in PurchaseMedicines)
            {
                float perPrice = med.getPrice() * med.getQuantity();
                total_price = total_price + perPrice;
            }
            return total_price;
        }

    }
```

BL Code: ExpiredMedicine.cs

```
class ExpiredMedicine
    {
        private string expiredmedicine;

        public string getExpiredmedicine() { return expiredmedicine; }
        public void setExpiredmedicine(string expiredmedicine) { 
this.expiredmedicine = expiredmedicine; }
        public ExpiredMedicine() { }
        public ExpiredMedicine(string expiredmedicine)
        {
            this.expiredmedicine = expiredmedicine;
        }
        public static int convertDatetodays(string date)
        {
            char[] datedays = { date[0], date[1], '\0' };
            char[] datemonths = { date[3], date[4], '\0' };
            char[] dateyears = { date[6], date[7], date[8], date[9], '\0' };
            string dateDays = new string(datedays);
            string dateMonths = new string(datemonths);
            string dateYears = new string(dateyears);
            return int.Parse(dateDays) + int.Parse(dateMonths) * 30 + 
(int.Parse(dateYears) - 2000) * 365;
        }
        public static bool isExpired(string currentDate, string expiryDate)
        {
            int currentDays = convertDatetodays(currentDate);
            int expiryDays = convertDatetodays(expiryDate);
```

```
                return (currentDays > expiryDays);
            }
        }
```

DL Code: MUserCRUD.cs

```
class MUser_CRUD
    {
       public static List<MUser> users = new List<MUser>();

        public static bool readData(string path)
        {
            if (File.Exists(path))
            {
                StreamReader fileVariable = new StreamReader(path);
                string record;
                while ((record = fileVariable.ReadLine()) != null)
                {
                    string[] line = record.Split(',');
                    string username = line[0];
                    string password = line[1];
                    string role = line[2];
                    MUser user = new MUser(username, password, role);
                    storeDataInList(user);
                }
                fileVariable.Close();
                return true;
            }
            return false;
        }
        public static void storeDataInList(MUser user)
        {
            users.Add(user);
        }

        public static void storeDataInFile(string path, MUser user)
        {
            StreamWriter file = new StreamWriter(path, true);
            file.WriteLine(user.getUsername() + "," + user.getPassword() + "," +
user.getRole());
            file.Flush();
            file.Close();
        }
        public static MUser signIn(MUser user)
        {
            foreach (MUser storedUser in users)
            {
                if (user.getUsername() == storedUser.getUsername() &&
user.getPassword() == storedUser.getPassword())
                {
                    return storedUser;
                }
            }
            return null;
        }
    }
```

DL Code: PatientCRUD.cs

```
class PatientCRUD
    {
        public static int bed = 150;
        public static List<Patient> patients = new List<Patient>();
        public static List<bool> bedAvaiables = new List<bool>();

        public static void Add_Patient(Patient p)
        {
            patients.Add(p);
        }
        public static bool addPatient_Dl(Patient p)
        {
            bool isbool;
            isbool = true;
            for (int i = 0; i < patients.Count; i++)
            {
                if (patients[i].getCnic() == p.getCnic())
                {
                    isbool = false;
                }
            }
            bed--;
            return isbool;
        }

        static string path_patient()
        {
            string path = "path\\Patient.txt";
            return path;
        }
        public static void save_patientdata(Patient p)
        {
            string path = path_patient();
            StreamWriter patient_file = new StreamWriter(path, true);

            patient_file.WriteLine(p.getName() + "," + p.getAge() + "," +
p.getContactno() + "," + p.getCnic() + "," + p.getAddress() + "," +
p.getVisitdate() + "," + p.getMedicalhistory());

            patient_file.Flush();

            patient_file.Close();

        }

        public static void load_patientdata()

        {

            string path = path_patient();

            string line;

            StreamReader patient_file = new StreamReader(path);

            while ((line = patient_file.ReadLine()) != null)

            {

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


                string name = record[0];

                int age = int.Parse(record[1]);

                string contactNo = record[2];

                string cnic = record[3];

                string address = record[4];

                string visit_date = record[5];

                string medicalHistory = record[6];

                Patient p = new Patient();

                p.setName(name);

                p.setAge(age);

                p.setContactno(contactNo);

                p.setCnic(cnic);

                p.setAddress(address);

                p.setVisitdate(visit_date);

                p.setMedicalhistory(medicalHistory);

                patients.Add(p);

            }

            patient_file.Close();

        }
        public static Patient update_patient_Dl(string patient_name, string
cnicno)

        {

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

            {

                if (patients[i].getName() == patient_name && patients[i].getCnic()
== cnicno)

                {

                    return patients[i];

                }


            }

            return null;

        }

       public static void update_patientdata()

        {

            string path = "path\\Patient.txt";

            StreamWriter patient_file = new StreamWriter(path);

            foreach (var Patient in patients)

            {

                patient_file.WriteLine(Patient.getName() + "," + Patient.getAge()
+ "," + Patient.getContactno() + "," + Patient.getCnic() + "," +
Patient.getAddress() + "," + Patient.getVisitdate() + "," +
Patient.getMedicalhistory());

            }

            patient_file.Flush();

            patient_file.Close();

        }


        public static void add_medicalReport(Patient p)

        {

            string path = "path\\medicalReport.txt";

            StreamWriter Report_file = new StreamWriter(path,true);

            Report_file.Write(p.getName() + ",");

            Report_file.Write(p.PatientReport.getLaboratorytest() + ",");

            if (p.PatientReport.getLaboratorytest() == "blood test")
            {
                Report_file.Write(p.PatientReport.getCholestrolLevel() + ",");
                Report_file.Write(p.PatientReport.getCholestrol() + ",");
                Report_file.Write(p.PatientReport.getTyphoid() + ",");
                Report_file.Write(p.PatientReport.getMalaria() + ",");
                Report_file.Write(p.PatientReport.getAnemia() + ",");
            }
            Report_file.Write(p.PatientReport.getDiagnostictest());
            Report_file.WriteLine();
            Report_file.Flush();
            Report_file.Close();
        }


        public static void load_medicalReport()
        {
            string Name, laboratory_test, Cholestrol, Typhoid, Malaria, Anemia, 
    diagnostic_test;
            int cholestrol_level;
            string line;
            string path = "path\\medicalReport.txt";
            int i = 0;
            StreamReader patient_file = new StreamReader(path);
            while ((line=patient_file.ReadLine())!=null)
            {
                string[] record = line.Split(',');
                Name = record[0];
                laboratory_test = record[1];
                cholestrol_level = int.Parse(record[2]);
                Cholestrol = record[3];
                Typhoid = record[4];
                Malaria = record[5];
                Anemia = record[6];
                diagnostic_test = record[7];
                patients[i].PatientReport.setLaboratorytest(laboratory_test);
                patients[i].PatientReport.setCholestrolLevel(cholestrol_level);
                patients[i].PatientReport.setCholestrol(Cholestrol);
                patients[i].PatientReport.setTyphoid(Typhoid);
                patients[i].PatientReport.setMalaria(Malaria);
                patients[i].PatientReport.setAnemia(Anemia);
                patients[i].PatientReport.setDiagnostictest(diagnostic_test);
                i++;
            }
            patient_file.Close();
        }
        public static bool dischargePatient(string discharge_patient, string 
dischargepatient_cnic)
        {
            bool discharge_check = false;
            for (int i = 0; i < patients.Count; i++)
            {
                if (patients[i].getCnic() == dischargepatient_cnic && 
discharge_patient == patients[i].getName())
                {
                    discharge_check = true;
                    patients.RemoveAt(i);
                    if (bed >= 1)
                    {
                        bed++;
                    }
                    break;
                }
            }
            return discharge_check;
        }

        public static void dataPatient_afterDelete()
        {
            string path = "path\\Patient.txt";
            StreamWriter patient_file = new StreamWriter(path);

            foreach (var Patient in patients)

            {

                patient_file.WriteLine(Patient.getName() + "," + Patient.getAge()
+ "," + Patient.getContactno() + "," + Patient.getCnic() + "," +
Patient.getAddress() + "," + Patient.getVisitdate() + "," +
Patient.getMedicalhistory());

            }

            patient_file.Flush();

            patient_file.Close();


        }

        public static void DataPatientReport_afterdelete()

        {

            string path = "path\\medicalReport.txt";

            StreamWriter Report_file = new StreamWriter(path);

            foreach (var Patient in patients)

            {

                Report_file.Write(Patient.getName() + ",");

                Report_file.Write(Patient.PatientReport.getLaboratorytest() +
",");

                if (Patient.PatientReport.getLaboratorytest() == "blood test")

                {

                    Report_file.Write(Patient.PatientReport.getCholestrolLevel() +
",");

                    Report_file.Write(Patient.PatientReport.getCholestrol() +
",");

                    Report_file.Write(Patient.PatientReport.getTyphoid() + ",");

                    Report_file.Write(Patient.PatientReport.getMalaria() + ",");

                    Report_file.Write(Patient.PatientReport.getAnemia() + ",");

                }

                Report_file.Write(Patient.PatientReport.getDiagnostictest());

                Report_file.WriteLine();

            }

            Report_file.Flush();

            Report_file.Close();
        }
        public static void dataPatientBill_afterdelete()
        {
            string path = "path\\Patient_bill.txt";
            StreamWriter billfile = new StreamWriter(path);
            foreach (var Patient in patients)
            {
                billfile.WriteLine(Patient.getName() + "," +
Patient.bill.getDaysPatientstayed() + "," + Patient.bill.getBedcharge() + "," +
Patient.bill.getLabtest() + "," + Patient.bill.getTotalbill());
            }
            billfile.Flush();
            billfile.Close();
        }
        public static void dataafter_deletefile()
        {
                dataPatient_afterDelete();
                dataPatientBill_afterdelete();
                DataPatientReport_afterdelete();
        }


        public static void add_billing(Patient p)
        {
            string path = "path\\Patient_bill.txt";
            StreamWriter billfile = new StreamWriter(path,true);
            billfile.WriteLine(p.getName() + "," + p.bill.getDaysPatientstayed() +
"," + p.bill.getBedcharge() + "," + p.bill.getLabtest() + "," +
p.bill.getTotalbill());
            billfile.Flush();
            billfile.Close();
        }
        public static void load_billing()
        {
            string patientnamebill;
            int dayspatientstayed;
            float bedCharge, lab_test, total_Bill;
            string line;
            string path = "path\\Patient_bill.txt";
            int countbill = 0;
            StreamReader file = new StreamReader(path);
            while ((line=file.ReadLine())!=null)
            {
                string[] record = line.Split(',');
                patientnamebill = record[0];
                dayspatientstayed = int.Parse(record[1]);
                bedCharge = float.Parse(record[2]);
                lab_test = float.Parse(record[3]);
                total_Bill = float.Parse(record[4]);

                //patients[countbill].PatientnameBill = patientnamebill;
                patients[countbill].bill.setDayspatientstayed(dayspatientstayed);
                patients[countbill].bill.setBedcharge(bedCharge);
                patients[countbill].bill.setLabtest(lab_test);
                patients[countbill].bill.setTotalbill(total_Bill);
                countbill++;
            }
            file.Close();
        }

        public static void saveBedChecker()
        {
            string path = "path\\BedAvailability.txt";
            StreamWriter file = new StreamWriter(path);
            for (int i = 0; i < bedAvaiables.Count; i++) {
                file.WriteLine(bedAvaiables[i]);
        }
            file.Flush();
            file.Close();
        }
```

```
        public static void loadBedChecker()

        {

            string path = "path\\BedAvailability.txt";

            string line;

            StreamReader file = new StreamReader(path);

            while ((line = file.ReadLine()) != null)

            {

                bool record = bool.Parse(line);

                bedAvaiables.Add(record);

            }

            file.Close();

        }

    }
```

DL Code: DoctorCRUD.cs

```
class Doctor_CRUD
    {
        public static List<Doctor> doctors = new List<Doctor>();

        public static void Add_Doctor(Doctor doctor)
        {
            doctors.Add(doctor);
        }
        public static void add_DoctorSchedule(Doctor doctor)
        {
            string path = "path\\doctor_schedule.txt";
            StreamWriter Doctor_schedule_file = new StreamWriter(path, true);
            Doctor_schedule_file.WriteLine(doctor.getScheduledate() + "," +
doctor.getDoctorname() + "," + doctor.getProfession() + "," +
doctor.getStartingtime() + "," + doctor.getEndingtime());
            Doctor_schedule_file.Flush();
            Doctor_schedule_file.Close();
        }
        public static void load_DoctorScheduleData()
        {
            string path = "path\\doctor_schedule.txt";
            string scheduledate, doctorname, Profession, starting_time,
ending_time;
            string line;
            StreamReader file = new StreamReader(path);
            while ((line = file.ReadLine()) != null)
            {
                string[] record = line.Split(',');
                scheduledate = record[0];
                doctorname = record[1];
                Profession = record[2];
                starting_time = record[3];
                ending_time = record[4];

               string Scheduledate = scheduledate;
               string Doctorname = doctorname;
               string profession = Profession;
               string startingtime = starting_time;
               string endingtime = ending_time;
               Doctor d = new Doctor();
                d.setScheduledate(Scheduledate);
                d.setDoctorname(Doctorname);
                d.setProfession(profession);
                d.setStartingtime(startingtime);
                d.setEndingtime(endingtime);
                doctors.Add(d);
            }
            file.Close();
        }

        public static void put_DoctorAppointment_inList(List<Appointment>
appointments)
        {
            foreach(var Doctor in doctors)
            {
                Doctor.AddData_inAppointmentList(appointments);
            }
        }


    }
```

DL Code: AppointmentCRUD.cs

```
class Appointment_CRUD
    {
        public static List<Appointment> appointments = new List<Appointment>();

        public static void Add_Appointment(Appointment appointment)
        {
            appointments.Add(appointment);
        }

       public static void canceledappointment(ref string cancelpatient)
        {
            for (int i = 0; i < appointments.Count; i++)
            {
                if (cancelpatient == appointments[i].getName())
                {

                    appointments.RemoveAt(i);
                    break;
                }

            }
        }
        public static void shiftedappointment(ref string ShiftedPatientname,ref 
string shiftdate,ref string shifttime)
        {
            for (int i = 0; i < appointments.Count; i++)
            {
                if (ShiftedPatientname == appointments[i].getName())
                {
                    appointments[i].setDateappoint(shiftdate);
                    appointments[i].setTimeappoint(shifttime);
                }
            }
        }
        public static void aftershiftedORcancelappointment(int num)
        {
            string path = "path\\appointment.txt";
            StreamWriter appointmentFile = new StreamWriter(path);
            appointmentFile.WriteLine(appointments[num].getName() + ","+ 
appointments[num].getDateappoint()+","+ appointments[num].getDoctor()+","+ 
appointments[num].getTimeappoint());
            appointmentFile.Flush();
            appointmentFile.Close();
        }
        public static void DATAaftershiftedORcancelappointment()
        {
            for (int i = 0; i < appointments.Count; i++)
            {
                aftershiftedORcancelappointment(i);
            }
        }

        public static void add_appointmentinFile(Appointment appointment)
        {
            string path = "path\\appointment.txt";
            StreamWriter appointmentFile = new StreamWriter(path, true);
            appointmentFile.WriteLine(appointment.getName() + "," + 
appointment.getDateappoint() + "," + appointment.getDoctor() + "," + 
appointment.getTimeappoint());
            appointmentFile.Flush();
            appointmentFile.Close();
        }
        public static void load_appointment()
        {
            string appointmentDoctor, appointmentPatientname, Dateappoint, 
Timeappoint;
            string line;
            string path = "path\\appointment.txt";
            StreamReader appointmentFile = new StreamReader(path);
            while ((line = appointmentFile.ReadLine()) != null)
            {
                string[] record = line.Split(',');
                appointmentPatientname = record[0];
                Dateappoint = record[1];
                appointmentDoctor = record[2];
                Timeappoint = record[3];

                string name = appointmentPatientname;
                string dateappoint = Dateappoint;
                string doctor = appointmentDoctor;
                string timeappoint = Timeappoint;
                Appointment ap = new Appointment(name, dateappoint, doctor, 
timeappoint);
                appointments.Add(ap);
                Doctor_CRUD.put_DoctorAppointment_inList(appointments);
            }
            appointmentFile.Close();
        }

    }
```

DL Code: MedicineCRUD.cs

```
class Medicine_CRUD
    {
        public static List<Medicine> meds = new List<Medicine>();
        public static void Addmedicine(Medicine med1)
        {
            meds.Add(med1);
        }
        public static string pathOfMedicineFile()
        {
            string path = "path\\medicine.txt";
            return path;
        }
        public static void addMedicineinFile()
        {
            string path = pathOfMedicineFile();
            StreamWriter medicineFile = new StreamWriter(path, true);
            medicineFile.WriteLine(meds[meds.Count - 1].getMedicinename() + "," + 
meds[meds.Count - 1].getQuantity() + "," + meds[meds.Count - 1].getDemandcheck() + 
"," + meds[meds.Count - 1].getExpirydate() + "," + meds[meds.Count - 
1].getPrice());
            medicineFile.Flush();
            medicineFile.Close();
        }

        public static void load_medicineFile()
        {
            string Addmedicine, Expirydate;
            int Quantity, Demandcheck;
            float Price;
            string path = pathOfMedicineFile();
            string record;
            StreamReader file = new StreamReader(path);
            while ((record = file.ReadLine()) != null)
            {
                string[] line = record.Split(',');
                Addmedicine = line[0];
                Quantity = int.Parse(line[1]);
                Demandcheck = int.Parse(line[2]);
                Expirydate = line[3];
                Price = float.Parse(line[4]);

                Medicine med1 = new Medicine();


                med1.setMedicinename(Addmedicine);
                med1.setQuantity(Quantity);
                med1.setDemandcheck(Demandcheck);
                med1.setExpirydate(Expirydate);
                med1.setPrice(Price);

                meds.Add(med1);
            }
            file.Close();
        }

        public static Medicine Updatemedicine_Check(string updatemedicine)
        {
            foreach (Medicine i in meds)
            {
                if (updatemedicine == i.getMedicinename())
                {
                    return i;
                }
            }
            return null;

        }
        public static void SaveChange()
        {
            string path = pathOfMedicineFile();
            StreamWriter medicineFile = new StreamWriter(path);
            foreach (var med in meds)
            {

                medicineFile.WriteLine(med.getMedicinename() + "," +
    med.getQuantity() + "," + med.getDemandcheck() + "," + med.getExpirydate() + "," +
    med.getPrice());
            }
            medicineFile.Flush();
            medicineFile.Close();
        }
        public static void saveUpdatedmedicine_data()
        {
            string path = pathOfMedicineFile();
            StreamWriter medicineFile = new StreamWriter(path);
            for (int i = 0; i < meds.Count; i++)
            {
                updatemedicinedata(i,medicineFile);
            }
            medicineFile.Flush();
            medicineFile.Close();
        }
        public static void updatemedicinedata(int num,StreamWriter medicineFile)
        {

            medicineFile.WriteLine(meds[num].getMedicinename() + "," +
    meds[num].getQuantity() + "," + meds[num].getDemandcheck() + "," +
    meds[num].getExpirydate() + "," + meds[num].getPrice());

        }

        public static bool deletemedicine(string delmedicine)
        {
            bool delmedicineCheck = false;
            for (int i = 0; i < meds.Count; i++)
            {
                if (delmedicine == meds[i].getMedicinename())
                {

                    delmedicineCheck = true;
                    meds.RemoveAt(i);
                    break;
                }
            }
            return delmedicineCheck;
        }

        public static void AFTERDeleteMedicine(int num,StreamWriter medicineFile)
        {

            medicineFile.WriteLine(meds[num].getMedicinename() + "," +
meds[num].getQuantity() + "," + meds[num].getDemandcheck() + "," +
meds[num].getExpirydate() + "," + meds[num].getPrice());

        }
        public static void DataAFTERDeleteMedicine()
        {
            string path = pathOfMedicineFile();
            StreamWriter medicineFile = new StreamWriter(path);
            for (int i = 0; i < meds.Count; i++)
            {
                AFTERDeleteMedicine(i,medicineFile);
            }
            medicineFile.Flush();
            medicineFile.Close();
        }
    }
```

DL Code: CustomerCRUD.cs

```
class Customer_CRUD
    {
        public static List<Customer> customers = new List<Customer>();

        public static void AddCustomerinList(Customer c)
        {
            customers.Add(c);
        }
        public static void add_purchaseMedicineinFile()
        {
            string path = "path\\purchaseMedicine.txt";
            StreamWriter purchasefile = new StreamWriter(path, true);
            purchasefile.Write(customers[customers.Count - 1].getcustomerName() +
":");
            List<Medicine> purchasemed = customers[customers.Count -
1].getPurchasedMedicines();
            for(int i = 0; i < purchasemed.Count - 1; i++)
            {
                purchasefile.Write(purchasemed[i].getMedicinename() + "," +
purchasemed[i].getQuantity() + "," + purchasemed[i].getPrice());
                purchasefile.Write("-");
            }
            purchasefile.Write(purchasemed[purchasemed.Count-1].getMedicinename()
+ "," + purchasemed[purchasemed.Count - 1].getQuantity() + "," +
purchasemed[purchasemed.Count - 1].getPrice());
            purchasefile.WriteLine();

            purchasefile.Flush();
            purchasefile.Close();
        }

        public static void load_purchaseMedicine()
        {
            string path = "path\\purchaseMedicine.txt";
            StreamReader purchasefile = new StreamReader(path);
            string line, Customername, purchasemedicine, Quantitypurchase,
total_Price;
            string Purchasedmedicines,separatePurchaseMed;

            while ((line = purchasefile.ReadLine()) != null)
            {
                Customer customer = new Customer();
                string[] record = line.Split(':');
                Customername = record[0];
                customer.setcustomerName(Customername);
                Purchasedmedicines = record[1];
                string[] record1 = Purchasedmedicines.Split('-');
                for(int i = 0; i < record1.Length; i++)
                {
                    separatePurchaseMed = record1[i];
                    string[] record3 = separatePurchaseMed.Split(',');
                    purchasemedicine = record3[0];
                    Quantitypurchase = record3[1];
                    total_Price = record3[2];
                    Medicine purchase_medicine = new Medicine();
                    purchase_medicine.setMedicinename(purchasemedicine);
                    purchase_medicine.setQuantity(int.Parse(Quantitypurchase));
                    purchase_medicine.setPrice(float.Parse(total_Price));
                    customer.AddMedicineInList(purchase_medicine);
                }
                customers.Add(customer);
            }

            purchasefile.Close();
        }

    }
```

DL Code: ExpiredMedicineCRUD.cs

```
class ExpiredMedicine_CRUD
    {
        public static List<ExpiredMedicine> expiredmed = new
List<ExpiredMedicine>();

        public static void addExpiredmedicine()
        {
            string path = "path\\ExpiredMedicine.txt";
```

```
                StreamWriter expirefile = new StreamWriter(path);
                for (int x = 0; x < expiredmed.Count; x++)
                {
                    expirefile.WriteLine(expiredmed[x].getExpiredmedicine());
                }
                expirefile.Flush();
                expirefile.Close();
            }
        }
```

Muser Forms code: Main Form.cs

```
    public partial class HMSApplication : Form
    {
        public HMSApplication()
        {
            InitializeComponent();


        }

        private void label1_Click(object sender, EventArgs e)
        {

        }

        private void button1_Click(object sender, EventArgs e)
        {
            Form newform = new SignUp_Form();
            newform.Show();
            this.Hide();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            Form newForm = new SignIn_Form();
            newForm.Show();
            this.Hide();
        }

        private void HMSApplication_Load(object sender, EventArgs e)
        {

        }

        private void button1_MouseEnter(object sender, EventArgs e)
        {
            button1.BackColor = Color.Black;
            button1.ForeColor = Color.White;
        }

        private void button1_MouseLeave(object sender, EventArgs e)
        {
            button1.BackColor = Color.FromArgb(128, 255, 255);
            button1.ForeColor = Color.Black;
        }

        private void button2_MouseEnter(object sender, EventArgs e)
        {
            button2.BackColor = Color.Black;
            button2.ForeColor = Color.White;
        }

        private void button2_MouseLeave(object sender, EventArgs e)
        {
            button2.BackColor = Color.FromArgb(128, 255, 255);
            button2.ForeColor = Color.Black;
        }
    }
```

Muser Forms code: SignUp Form.cs

```
    public partial class SignUp_Form : Form
    {
        public SignUp_Form()
        {
            InitializeComponent();
        }
        private void ClearDataFromForm()
        {
            txtusername.Text = "";
            txtpassword.Text = "";
            txtrole.Text = "";
        }
        private void SignUp_Form_Load(object sender, EventArgs e)
        {
        }

        private void button1_Click(object sender, EventArgs e)
        {
            string username = txtusername.Text;
            string password = txtpassword.Text;
            string role = txtrole.Text;
            string path = "data.txt";
            MUser user = new MUser(username, password, role);
            MUser_CRUD.addUserIntoList(user);
            MUser_CRUD.storeIntoFile(path, user);
            MessageBox.Show("User Added Successfully");
            ClearDataFromForm();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            Form newform = new HMSApplication();
            newform.Show();
            this.Hide();
        }

        private void chkboxshowpasscode_CheckedChanged(object sender, EventArgs e)
        {
            if (chkboxshowpasscode.Checked)
            {
                txtpassword.UseSystemPasswordChar = true;
            }
            else
            {
                txtpassword.UseSystemPasswordChar = false;
            }
        }
        private void button2_MouseEnter(object sender, EventArgs e)
        {
            button2.BackColor = Color.Black;
            button2.ForeColor = Color.White;
        }

        private void button2_MouseLeave(object sender, EventArgs e)
        {
            button2.BackColor = Color.FromArgb(128, 255, 255);
            button2.ForeColor = Color.Black;
        }

        private void button1_MouseEnter(object sender, EventArgs e)
        {
            button1.BackColor = Color.Black;
            button1.ForeColor = Color.White;
        }

        private void button1_MouseLeave(object sender, EventArgs e)
        {
            button1.BackColor = Color.FromArgb(128, 255, 255);
            button1.ForeColor = Color.Black;
        }
    }
```

Muser Forms code: SignIn Form.cs

```
    public partial class SignIn_Form : Form
    {
        public SignIn_Form()
        {
            InitializeComponent();
        }
        private void ClearDataFromForm()
        {
            txtusername.Text = "";
            txtpassword.Text = "";
        }
        private void SignIn_Form_Load(object sender, EventArgs e)
        {

        }

        private void SignInbtn_Click(object sender, EventArgs e)
        {
            string username = txtusername.Text;
            string password = txtpassword.Text;
            MUser user = new MUser(username, password);
            MUser validUser = MUser_CRUD.SignIn(user);
            if (validUser != null)
            {
                if (validUser.UserRole == "Admin" || validUser.UserRole == "admin")
                {
                    Form newform = new Admin_menu_Form();
                    newform.Show();

                }else if (validUser.UserRole == "patient" || validUser.UserRole ==
    "Patient")
                {
                    Form newform = new Patient_menu_form();
                    newform.Show();
                }else if(validUser.UserRole == "Pharmacist" || validUser.UserRole ==
"pharmacist") {
                    Form newform = new Pharmacy_Menu_Form();
                    newform.Show();
                }
                this.Hide();
            }
            else
            {
                MessageBox.Show("User is Invalid");
            }
            ClearDataFromForm();
        }

        private void Back_Click(object sender, EventArgs e)
        {
            Form newform = new HMSApplication();
            newform.Show();
            this.Hide();
        }

        private void chkboxShowPasscode_CheckedChanged(object sender, EventArgs e)
        {
            if (chkboxShowPasscode.Checked)
            {
                txtpassword.UseSystemPasswordChar = true;
            }
            else
            {
                txtpassword.UseSystemPasswordChar = false;
            }
        }

        private void Back_MouseEnter(object sender, EventArgs e)
        {
            Back.BackColor = Color.Black;
            Back.ForeColor = Color.White;
        }

        private void Back_MouseLeave(object sender, EventArgs e)
        {
            Back.BackColor = Color.FromArgb(128, 255, 255);
            Back.ForeColor = Color.Black;
        }

        private void SignInbtn_MouseEnter(object sender, EventArgs e)
        {
            SignInbtn.BackColor = Color.Black;
            SignInbtn.ForeColor = Color.White;
        }

        private void SignInbtn_MouseLeave(object sender, EventArgs e)
        {
            SignInbtn.BackColor = Color.FromArgb(128, 255, 255);
            SignInbtn.ForeColor = Color.Black;
        }
    }
```

Patient Forms code: Add Patient.cs

```
public partial class Add_Patient_Form : Form
    {
        public Add_Patient_Form()
        {
            InitializeComponent();
        }

        private void label1_Click(object sender, EventArgs e)
        {

        }

        private void btnAdd_Click(object sender, EventArgs e)
        {
            string patient_name = txtpatient.Text;
            int age = int.Parse(txtage.Text);
            string phoneno = txtphone.Text;
            string cnic = txtcnicno.Text;
            string address = txtaddress.Text;
            DateTime date = dtvisitdate.Value;
            string datevisit = Convert.ToDateTime(date).Date.ToString("d");
            string medicalHistory = txtmedicalhistory.Text;
            Patient p = new Patient();
            p.setName(patient_name);
            p.setAge(age);
            p.setContactno(phoneno);
            p.setCnic(cnic);
            p.setAddress(address);
            p.setVisitdate(datevisit);
            p.setMedicalhistory(medicalHistory);
            ADD_Patient(p);

        }
        private void ADD_Patient(Patient P)
        {
            bool check = PatientCRUD.addPatient_Dl(P);
            if (check == false)
            {
               MessageBox.Show("This patient record is already exist");
                Empty();
            }
            else
            {
                PatientCRUD.Add_Patient(P);
                PatientCRUD.save_patientdata(P);
                MessageBox.Show("The Patient is Added Successfully and Please give
proper info by choosing other options for single Patient before adding next Patient");
                this.Close();
                Form newform = new Admin_menu_Form();
                newform.Show();
            }
        }
        private void Empty()
        {
            txtpatient.Text = String.Empty;
            txtage.Text = String.Empty;
            txtcnicno.Text = String.Empty;
            txtphone.Text = String.Empty;
            txtaddress.Text = String.Empty;
            dtvisitdate.Value = DateTime.Now;
        }
        private void Add_Patient_Form_Load(object sender, EventArgs e)
        {

        }

        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form newform = new Admin_menu_Form();
            newform.Show();
            this.Hide();
        }

        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }

        private void btnAdd_MouseEnter(object sender, EventArgs e)
        {
            btnAdd.BackColor = Color.Black;
            btnAdd.ForeColor = Color.White;
        }

        private void btnAdd_MouseLeave(object sender, EventArgs e)
        {
            btnAdd.BackColor=Color.FromArgb(128, 255, 255);
            btnAdd.ForeColor=Color.Black;
        }
    }
```

Patient Forms code: Update Patient.cs

```
    public partial class Update_Patient_Form : Form
    {
        Patient p1;
        DataTable table = new DataTable("UpdatePatient");
        public Update_Patient_Form()
        {
            InitializeComponent();
        }

        private void Update_Patient_Form_Load(object sender, EventArgs e)
        {

        }

        private void btnCheck_Click(object sender, EventArgs e)
        {
            string patientName = txtPatientUpdatename.Text;
            string cnicNO = txtCnicnoUpdate.Text;
            UPDATE_Patient_Check(ref patientName,ref cnicNO);
        }
        private void UPDATE_Patient_Check(ref string patientName,ref string cnicNO)
        {
            p1 = PatientCRUD.update_patient_Dl(patientName, cnicNO);
            ifPatientnotExist(p1);
        }
        private void PatientColumnsHeading()
        {
            table.Columns.Add("Patient name", Type.GetType("System.String"));
            table.Columns.Add("Age", Type.GetType("System.String"));
            table.Columns.Add("Phone no", Type.GetType("System.String"));
            table.Columns.Add("Cnic", Type.GetType("System.String"));
            table.Columns.Add("Address", Type.GetType("System.String"));
        }
        private void ifPatientnotExist(Patient p)
        {
            if (p != null)
            {
                PatientColumnsHeading();
                table.Rows.Add(p.getName(), p.getAge(), p.getContactno(), p.getCnic(), 
p.getAddress());
                UpdatePatientGV.DataSource = table;
                txtPatientname.Text=p.getName();
                txtage.Text=""+p.getAge();
                txtPhoneno.Text=p.getContactno();
                txtCnicno.Text=p.getCnic();
                string date = p.getVisitdate();
                dtvisitdate.Value=DateTime.Parse(date);
                txtAddress.Text=p.getAddress();
                txtmedicalHistory.Text=p.getMedicalhistory();
                /* Admin_interface.updatepatient_after(p);
                 PatientCRUD.update_patientdata();*/
            }
            if(p == null)
            {
                MessageBox.Show("The patient is not exist in our database");
                empty();
            }
        }

        private void btnUpdate_Click(object sender, EventArgs e)
        {
            string name = txtPatientname.Text;
            int age = int.Parse(txtage.Text);
            string phoneNo = txtPhoneno.Text;
            string cnic = txtCnicno.Text;
            DateTime date = dtvisitdate.Value;
            string datevisit = Convert.ToDateTime(date).Date.ToString("d");
            string Address = txtAddress.Text;
            string MedicalHistory = txtmedicalHistory.Text;
            p1.setName(name);
            p1.setAge(age);
            p1.setContactno(phoneNo);
            p1.setCnic(cnic);
            p1.setVisitdate(datevisit);
            p1.setAddress(Address);
            p1.setMedicalhistory(MedicalHistory);
            UpdateMessage();
            empty();
        }
        private void UpdateMessage()
        {
            MessageBox.Show("Selected Patient data is Updated");
        }
        private void empty()
        {
            txtPatientUpdatename.Text=String.Empty;
            txtCnicnoUpdate.Text=String.Empty;
            txtPatientname.Text = String.Empty;
```

```
                txtAddress.Text = String.Empty;
                txtage.Text = String.Empty;
                txtCnicno.Text = String.Empty;
                txtmedicalHistory.Text = String.Empty;
                txtPhoneno.Text = String.Empty;
                dtvisitdate.Value = DateTime.Now;
                UpdatePatientGV.DataSource = null;
                table.Columns.Clear();
                table.Rows.Clear();
            }

            private void HometoolStripMenuItem1_Click(object sender, EventArgs e)
            {
                Form newform = new Admin_menu_Form();
                newform.Show();
                this.Hide();
            }

            private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
            {
                Environment.Exit(1);
            }

            private void btnCheck_MouseEnter(object sender, EventArgs e)
            {
                btnCheck.BackColor = Color.Black;
                btnCheck.ForeColor = Color.White;
            }

            private void btnCheck_MouseLeave(object sender, EventArgs e)
            {
                btnCheck.BackColor = Color.FromArgb(128, 255, 255);
                btnCheck.ForeColor = Color.Black;
            }

            private void btnUpdate_MouseEnter(object sender, EventArgs e)
            {
                btnUpdate.BackColor = Color.Black;
                btnUpdate.ForeColor = Color.White;
            }

            private void btnUpdate_MouseLeave(object sender, EventArgs e)
            {
                btnUpdate.BackColor = Color.FromArgb(128, 255, 255);
                btnUpdate.ForeColor = Color.Black;
            }
        }
```

Patient Forms code: View Patient.cs

```
    public partial class View_Patient_form : Form
        {
            private string path = "path\\Patient.txt";
            DataTable table = new DataTable("PatientTable");
            public View_Patient_form()
            {
                InitializeComponent();
            }

            private void tableLayoutPanel1_Paint(object sender, PaintEventArgs e)
            {

            }

            private void dataGridView1_CellContentClick(object sender,
    DataGridViewCellEventArgs e)
            {

            }

            private void View_Patient_form_Load(object sender, EventArgs e)
            {
                table.Columns.Add("Patientname", Type.GetType("System.String"));
                table.Columns.Add("Age", Type.GetType("System.String"));
                table.Columns.Add("Phone no", Type.GetType("System.String"));
                table.Columns.Add("Cnic", Type.GetType("System.String"));
                table.Columns.Add("Address", Type.GetType("System.String"));

                for (int i = 0; i < PatientCRUD.patients.Count; i++)
                {
                    Patient p = PatientCRUD.patients[i];

    table.Rows.Add(p.getName(),p.getAge(),p.getContactno(),p.getCnic(),p.getAddress());
                }
                PatientGV.DataSource = table;
            }

            private void HometoolStripMenuItem1_Click(object sender, EventArgs e)
            {
                Form newform = new Admin_menu_Form();
                newform.Show();
                this.Hide();
            }

            private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
            {
                Environment.Exit(1);
            }

            private void txtSearchBox_TextChanged(object sender, EventArgs e)
            {
                DataView dvPatientTable = table.DefaultView;
                dvPatientTable.RowFilter = "Patientname Like '%" + txtSearchBox.Text +
    "%'";
            }

            private void txtSearchBox_Enter(object sender, EventArgs e)
            {
                txtCnicSearch.Text = "Search By Cnic";
                txtCnicSearch.ForeColor = Color.DimGray;
                if (txtSearchBox.Text=="Search By Name")
                {
                    txtSearchBox.Text = "";
                    txtSearchBox.ForeColor =Color.Black;

                }
            }

            private void txtSearchBox_Leave(object sender, EventArgs e)
            {
                if (txtSearchBox.Text == "")
                {
                    txtSearchBox.Text = "Search By Name";
                    txtSearchBox.ForeColor = Color.DimGray;
                }
            }

            private void txtCnicSearch_TextChanged(object sender, EventArgs e)
            {
                DataView dvPatientTable1 = table.DefaultView;
                dvPatientTable1.RowFilter = "Cnic Like '%" + txtCnicSearch.Text + "%'";
            }

            private void txtCnicSearch_Enter(object sender, EventArgs e)
            {
                txtSearchBox.Text = "Search By Name";
                txtSearchBox.ForeColor = Color.DimGray;
                if (txtCnicSearch.Text == "Search By Cnic")
                {
                    txtCnicSearch.Text = "";
                    txtCnicSearch.ForeColor = Color.Black;

                }
            }

            private void txtCnicSearch_Leave(object sender, EventArgs e)
            {
                if (txtCnicSearch.Text == "")
                {
                    txtCnicSearch.Text = "Search By Cnic";
                    txtCnicSearch.ForeColor = Color.DimGray;
                }
            }

            private void label2_Click(object sender, EventArgs e)
            {

            }
        }
```

Patient forms code: Discharge Patient.cs

```
    public partial class Discharge_Patient_Form : Form
    {
        public Discharge_Patient_Form()
        {
            InitializeComponent();
        }

        private void btnDelete_Click(object sender, EventArgs e)
        {
            string discharge_patient = txtname.Text;
            string dischargepatient_cnic = txtCnic.Text;
            bool discharge_check = PatientCRUD.dischargePatient(discharge_patient,
    dischargepatient_cnic);
            if (discharge_check == true)
            {
                MessageBox.Show("The Patient is discharge successfully");
                PatientCRUD.dataafter_deletefile();
            }
            else
            {
                MessageBox.Show("The patient does not exist in our database");
            }
            empty();
        }
        private void empty()
        {
            txtname.Text = null;
            txtCnic.Text = null;
        }


        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form newform = new Admin_menu_Form();
            newform.Show();
            this.Hide();
        }

        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }

        private void btnDelete_MouseEnter(object sender, EventArgs e)
        {
            btnDelete.BackColor = Color.Black;
            btnDelete.ForeColor = Color.White;
        }

        private void btnDelete_MouseLeave(object sender, EventArgs e)
        {
            btnDelete.BackColor = Color.FromArgb(128, 255, 255);
            btnDelete.ForeColor = Color.Black;
        }
    }
```

Admin Forms code: Suggested Medicine.cs

```
    public partial class Suggest_Medicine_Form : Form
    {
        string[] symptoms = { " ", " ", " ", " ", " ", " " };
        char check1 = ' ', check2 = ' ', check3 = ' ';
        Patient P = PatientCRUD.patients[PatientCRUD.patients.Count - 1];
        int count = 0;
        public Suggest_Medicine_Form()
        {
            InitializeComponent();
        }

        private void btnNext_Click(object sender, EventArgs e)
        {
            string Medicine = P.GiveMedicine(check1, check2, check3);
            MessageBox.Show("Suggested medicines for you is "+Medicine);
            this.Close();
            Form newform = new Admin_menu_Form();
            newform.Show();
        }

        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form newform = new Admin_menu_Form();
            newform.Show();
            this.Hide();
        }

        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }

        private void btnAdd_MouseEnter(object sender, EventArgs e)
        {
            btnAdd.BackColor = Color.Black;
            btnAdd.ForeColor = Color.White;
        }

        private void btnAdd_MouseLeave(object sender, EventArgs e)
        {
            btnAdd.BackColor = Color.FromArgb(128, 255, 255);
            btnAdd.ForeColor = Color.Black;
        }

        private void btnNext_MouseEnter(object sender, EventArgs e)
        {
            btnNext.BackColor = Color.Black;
            btnNext.ForeColor = Color.White;
        }

        private void btnNext_MouseLeave(object sender, EventArgs e)
        {
            btnNext.BackColor = Color.FromArgb(128, 255, 255);
            btnNext.ForeColor = Color.Black;
        }

        private void btnAdd_Click(object sender, EventArgs e)
        {
            if (count < 6)
            {
                symptoms[count] = txtSymptom.Text;
                P.SymptomsCheck(symptoms, count, ref check1, ref check2, ref check3);
                count++;
                empty();
            }
            else
            {
                MessageBox.Show("You have entered all the symptoms");
            }
        }
        private void empty()
        {
            txtSymptom.Text = String.Empty;
        }
    }
```

Admin Forms code: Generate Report.cs

```
    public partial class Generate_medical_Report_Form : Form
    {
        Patient p = PatientCRUD.patients[PatientCRUD.patients.Count - 1];
        public Generate_medical_Report_Form()
        {
            InitializeComponent();
        }

        private void label3_Click(object sender, EventArgs e)
        {

        }
        private void btnNext_Click(object sender, EventArgs e)
        {
            string LabTestName = txtLabtestName.Text;
            int cholestrolLevel = int.Parse(txtCholestrolLevel.Text);
            string Typhoid = txtTyphoid.Text;
            string Malaria = txtMalaria.Text;
            string Anemia = txtAnemia.Text;
            string DiagnosticTestname = txtDiagnosticTest.Text;
                p.PatientReport.setLaboratorytest(LabTestName);
            if (LabTestName == "blood test")
            {
                p.PatientReport.setCholestrolLevel(cholestrolLevel);
                p.PatientReport.Generatemedicalreport();
                p.PatientReport.setTyphoid(Typhoid);
                p.PatientReport.setMalaria(Malaria);
                p.PatientReport.setAnemia(Anemia);
            }
            p.PatientReport.setDiagnostictest(DiagnosticTestname);
            PatientCRUD.add_medicalReport(p);
            MessageBox.Show("Report is generated successfully");
            this.Close();
            Form newform = new Admin_menu_Form();
            newform.Show();
        }

        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form newform = new Admin_menu_Form();
            newform.Show();
            this.Hide();
        }

        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }

        private void btnNext_MouseEnter(object sender, EventArgs e)
        {
            btnNext.BackColor = Color.Black;
            btnNext.ForeColor = Color.White;
        }

        private void btnNext_MouseLeave(object sender, EventArgs e)
        {
            btnNext.BackColor = Color.FromArgb(128, 255, 255);
            btnNext.ForeColor = Color.Black;
        }
    }
```

Admin Forms code: Billing/Invoicing.cs

```
public partial class Billing_Form : Form
    {
        int count = 1;
        int firstCharge = 0;
        int secondCharge = 0;
        Patient P = PatientCRUD.patients[PatientCRUD.patients.Count - 1];
        PatientBill bill = PatientCRUD.patients[PatientCRUD.patients.Count-1].bill;
        public Billing_Form()
        {
            InitializeComponent();
        }

        private void btnCalculatebill_Click(object sender, EventArgs e)
        {
            double totalpricemedicine = 0F;
            int days = Convert.ToInt32(Math.Round(numpatientdaysinhospital.Value, 0));
            bill.setDayspatientstayed(days);
            for (int i = 0; i < Customer_CRUD.customers.Count; i++)
            {
                if (P.getName() == Customer_CRUD.customers[i].getcustomerName())
                {
                    totalpricemedicine = Customer_CRUD.customers[i].Calculate_Bill();
                    break;
                }
            }
            int bedCharge = Convert.ToInt32(Math.Round(numunitdaycharge.Value, 0));
            bill.setBedcharge(bedCharge);
            int labtest = Convert.ToInt32(Math.Round(numlabbill.Value, 0)); ;
            bill.setLabtest(labtest);
            bill.calculate_totalBill(firstCharge, secondCharge, bill.getBedcharge(),
totalpricemedicine);
            double totalbill = bill.getTotalbill();
            txttotalBill.Text = ""+totalbill;
            PatientCRUD.add_billing(P);
        }

        private void btnAdd_Click(object sender, EventArgs e)
        {
            string service = txtService.Text;
            if (count < 3)
            {
                if (count == 1)
                {
                    bill.setServices(service);
                    firstCharge = bill.ServiceCheck();
                }
                if (count == 2)
                {
                    if (service != bill.getServices())
                    {
                        bill.setServices(service);
                        secondCharge = bill.ServiceCheck();
                    }
                    else
                    {
                        MessageBox.Show("You Already added this service");
                        count -= 1;
                    }
                }
            }
```

```
                else { MessageBox. Show("You have added all available services NO MORE");
      }
                count++;
            }


            private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
            {
                Form newform = new Admin_menu_Form();
                newform.Show();
                this.Hide();
            }

            private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
            {
                Environment.Exit(1);
            }

            private void btnAdd_MouseEnter(object sender, EventArgs e)
            {
                btnAdd.BackColor = Color.Black;
                btnAdd.ForeColor = Color.White;
            }

            private void btnAdd_MouseLeave(object sender, EventArgs e)
            {
                btnAdd.BackColor = Color.FromArgb(128, 255, 255);
                btnAdd.ForeColor = Color.Black;
            }

            private void btnCalculatebill_MouseEnter(object sender, EventArgs e)
            {
                btnCalculatebill.BackColor = Color.Black;
                btnCalculatebill.ForeColor = Color.White;
            }

            private void btnCalculatebill_MouseLeave(object sender, EventArgs e)
            {
                btnCalculatebill.BackColor = Color.FromArgb(128, 255, 255);
                btnCalculatebill.ForeColor = Color.Black;
            }
        }
```

Doctor Forms code: Set DoctorSchedule.cs

```
    public partial class Manage_DoctorSchedule_Form : Form
        {
            public Manage_DoctorSchedule_Form()
            {
                InitializeComponent();
            }

            private void btnAdd_Click(object sender, EventArgs e)
            {
                DateTime date = dtDoctordate.Value;
                string Doctordate= Convert.ToDateTime(date).Date.ToString("d");
                string DoctorName = txtDoctorname.Text;
                string profession = txtDoctorSpecialty.Text;
                string startingTime = txtStartingTime.Text;
                string endingTime = txtEndingTime.Text;
                Doctor d = new Doctor();
                d.setScheduledate(Doctordate);
                d.setDoctorname(DoctorName);
                d.setProfession(profession);
                d.setStartingtime(startingTime);
                d.setEndingtime(endingTime);
                Doctor_CRUD.Add_Doctor(d);
                Doctor_CRUD.add_DoctorSchedule(d);
                ShowDialogbox();
            }
            private void ShowDialogbox()
            {
                DialogResult res = MessageBox.Show("Do you want to add another Doctor
    Schedule", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
                if (res == DialogResult.Yes)
                {
                    empty();
                }
                else
                {
                    this.Close();
                }
            }
            private void empty()
            {
                txtDoctorname.Text = String.Empty;
                txtDoctorSpecialty.Text = String.Empty;
                txtEndingTime.Text = String.Empty;
                txtStartingTime.Text = String.Empty;
                dtDoctordate.Value = DateTime.Now;
            }


            private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
            {
                Form newform = new Admin_menu_Form();
                newform.Show();
                this.Hide();
            }

            private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
            {
                Environment.Exit(1);
            }

            private void btnAdd_MouseEnter(object sender, EventArgs e)
            {
                btnAdd.BackColor = Color.Black;
                btnAdd.ForeColor = Color.White;
            }

            private void btnAdd_MouseLeave(object sender, EventArgs e)
            {
                btnAdd.BackColor = Color.FromArgb(128, 255, 255);
                btnAdd.ForeColor = Color.Black;
            }
        }
```

Admin Forms code: Manage Doctor Appointment.cs

```
    public partial class ManageAppointments_form : Form
        {
            private DataTable appointmentTable = new DataTable("AppointmentTable");

            public ManageAppointments_form()
            {
                InitializeComponent();
            }

            private void ManageAppointments_form_Load(object sender, EventArgs e)
            {
                TableColumnsHeading();
                ShowAppointments();
                table_fun();
            }
            public void TableColumnsHeading()
            {
                appointmentTable.Columns.Clear();
                appointmentTable.Rows.Clear();
                appointmentTable.Columns.Add("Patient name",
    Type.GetType("System.String"));
                appointmentTable.Columns.Add("Appointment Date",
    Type.GetType("System.String"));
                appointmentTable.Columns.Add("Appointment Doctor",
    Type.GetType("System.String"));
                appointmentTable.Columns.Add("Appointment Time",
    Type.GetType("System.String"));
            }
            public void ShowAppointments()
            {
                for (int i = 0; i < Appointment_CRUD.appointments.Count; i++)
                {
                    Appointment appointment = Appointment_CRUD.appointments[i];
                    appointmentTable.Rows.Add(appointment.getName(),
    appointment.getDateappoint(), appointment.getDoctor(), appointment.getTimeappoint());
                }


            }
            public void table_fun()
            {
                AppointmentsGV.DataSource = appointmentTable;
            }
            public static void t()
            {
                ManageAppointments_form appointments_Form = new ManageAppointments_form();
                appointments_Form.table_fun();
            }
            private void btnShiftAppointment_Click(object sender, EventArgs e)
            {
                Form newform = new Patient_Form.ShiftAppointment_Form();
                newform.Tag = this;
                newform.Show();

            }

            private void btnCancelAppointment_Click(object sender, EventArgs e)
            {
                Form newform = new Patient_Form.CancelAppointment_Form();
                newform.Tag = this;
                newform.Show();
            }



            private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
            {
                Form newform = new Admin_menu_Form();
                newform.Show();
                this.Hide();
            }

            private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
            {
                Environment.Exit(1);
            }

            private void btnShiftAppointment_MouseEnter(object sender, EventArgs e)
            {
                btnShiftAppointment.BackColor = Color.Black;
                btnShiftAppointment.ForeColor = Color.White;
            }

            private void btnShiftAppointment_MouseLeave(object sender, EventArgs e)
            {
                btnShiftAppointment.BackColor = Color.FromArgb(128, 255, 255);
                btnShiftAppointment.ForeColor = Color.Black;
            }

            private void btnCancelAppointment_MouseEnter(object sender, EventArgs e)
            {
                btnCancelAppointment.BackColor = Color.Black;
                btnCancelAppointment.ForeColor = Color.White;
            }

            private void btnCancelAppointment_MouseLeave(object sender, EventArgs e)
            {
                btnCancelAppointment.BackColor = Color.FromArgb(128, 255, 255);
                btnCancelAppointment.ForeColor = Color.Black;
            }
        }
```

Shift Appointment form:

```
    public partial class ShiftAppointment_Form : Form
        {
            public ShiftAppointment_Form()
            {
                InitializeComponent();
            }

            private void btnShift_Click(object sender, EventArgs e)
            {
                string ShiftedPatientname = txtname.Text;
                DateTime date = dtDateAppointment.Value;
                string shiftdate = Convert.ToDateTime(date).Date.ToString("d");
                string shiftTime = txttime.Text;
                bool exist = Appointment_CRUD.shiftedappointment(ref ShiftedPatientname,
    ref shiftdate, ref shiftTime);
```

```
            if (exist == true) { MessageBox.Show("Selected Appointment is Shifted 
Successfully"); }
            else { MessageBox.Show("This Patient has no appointment write correct 
patient"); }
            Appointment_CRUD.DATAaftershiftedORcancelappointment();
            ManageAppointments_form nform = this.Tag as ManageAppointments_form;
            nform.TableColumnsHeading();
            nform.ShowAppointments();
            this.Close();
        }

        private void exitToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }

        private void btnShift_MouseEnter(object sender, EventArgs e)
        {
            btnShift.BackColor = Color.Black;
            btnShift.ForeColor = Color.White;
        }

        private void btnShift_MouseLeave(object sender, EventArgs e)
        {
            btnShift.BackColor = Color.FromArgb(128, 255, 255);
            btnShift.ForeColor = Color.Black;
        }
    }
```

Cancel Appointment Form:

```
public partial class CancelAppointment_Form : Form
    {
        public CancelAppointment_Form()
        {
            InitializeComponent();
        }

        private void btnCancel_Click(object sender, EventArgs e)
        {
            string cancelpatient = txtCancelAppointmentPatient.Text;
            bool exist = Appointment_CRUD.canceledappointment(ref cancelpatient);
            if (exist == true) { MessageBox.Show("The Appointment of this patient is 
Canceled"); }
            else { MessageBox.Show("This Patient has no Appointment"); }
            Appointment_CRUD.cleanFile();
            Appointment_CRUD.DATAaftershiftedORcancelappointment();
            ManageAppointments_form nform = this.Tag as ManageAppointments_form;
            nform.TableColumnsHeading();
            nform.ShowAppointments();
            this.Close();
        }

        private void exitToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }

        private void btnCancel_MouseEnter(object sender, EventArgs e)
        {
            btnCancel.BackColor = Color.Black;
            btnCancel.ForeColor = Color.White;
        }

        private void btnCancel_MouseLeave(object sender, EventArgs e)
        {
            btnCancel.BackColor = Color.FromArgb(128, 255, 255);
            btnCancel.ForeColor = Color.Black;
        }
    }
```

Admin Forms code: Check Doctor Appointment.cs

```
public partial class Check_Doctor_Appointment_Form : Form
    {
        DataTable Doctors = new DataTable("DoctorTable");
        public Check_Doctor_Appointment_Form()
        {

            InitializeComponent();
        }

        private void btnCheck_Click(object sender, EventArgs e)
        {

            DateTime date = dtDate.Value;
            string Date = Convert.ToDateTime(date).Date.ToString("d");
            string Doctorname = txtDoctorname.Text;
            TablecolumnsHeading();
            foreach (var doctor in Doctor_CRUD.doctors)
            {
                if (Doctorname == doctor.getDoctorname() && Date ==
doctor.getScheduledate())
                {

                    for (int i = 0; i < doctor.myAppointments.Count; i++)
                    {
                        Appointment a = doctor.myAppointments[i];
                        Doctors.Rows.Add(a.getName(), a.getTimeappoint());
                    }
                    break;
                }
            }
            DoctorsGV.DataSource = Doctors;
        }
        private void TablecolumnsHeading()
        {
            Doctors.Columns.Add("Patient name", Type.GetType("System.String"));
            Doctors.Columns.Add("Appointment Time", Type.GetType("System.String"));
        }



        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form newform = new Admin_menu_Form();
            newform.Show();
            this.Hide();
        }
        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }

        private void btnCheck_MouseEnter(object sender, EventArgs e)
        {
            btnCheck.BackColor = Color.Black;
            btnCheck.ForeColor = Color.White;
        }

        private void btnCheck_MouseLeave(object sender, EventArgs e)
        {
            btnCheck.BackColor = Color.FromArgb(128, 255, 255);
            btnCheck.ForeColor = Color.Black;
        }
    }
```

Individual PatientData Forms code: Detail.cs

```
public partial class DetailForm : Form
    {
       public static string Patientname;
       public static string patientCnic;
        public DetailForm()
        {
            InitializeComponent();
        }

        private void btnnext_Click(object sender, EventArgs e)
        {
            Patientname = txtpatient.Text;
            patientCnic = txtcnicno.Text;
            bool checkPatient = false;
            for (int i = 0; i < PatientCRUD.patients.Count; i++)
            {
                if (patientCnic == PatientCRUD.patients[i].getCnic())
                {
                    checkPatient = true;
                    break;
                }
                else { checkPatient = false; }
            }
            if(checkPatient == true) { this.Close(); Form newform = new
Patient_menu_form();  newform.Show();}
            if(checkPatient == false) { MessageBox.Show("This patient is not store in
our database"); empty(); }
        }
        private void empty()
        {
            txtcnicno.Text = null;
            txtpatient.Text = null;
        }

        private void btnBack_Click(object sender, EventArgs e)
        {
            Form newform = new Patient_Form.Patient_menu_form();
            newform.Show();
            this.Hide();
        }

        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form newform = new Patient_menu_form();
            newform.Show();
            this.Hide();
        }

        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }

        private void btnnext_MouseEnter(object sender, EventArgs e)
        {
            btnnext.BackColor = Color.Black;
            btnnext.ForeColor = Color.White;
        }

        private void btnnext_MouseLeave(object sender, EventArgs e)
        {
            btnnext.BackColor = Color.FromArgb(128, 255, 128);
            btnnext.ForeColor = Color.Black;
        }
    }
```

Individual PatientData Forms code: View Prescription.cs

```
public partial class ViewPrescriptionForm : Form
    {
        public ViewPrescriptionForm()
        {
            InitializeComponent();
        }

        private void ViewPrescriptionForm_Load(object sender, EventArgs e)
        {

        }

        private void ViewPrescriptionForm_Load_1(object sender, EventArgs e)
        {
            for (int i = 0; i < PatientCRUD.patients.Count; i++)
            {
                if (DetailForm.patientCnic == PatientCRUD.patients[i].getCnic())
                {
                    lblprescription.Text = PatientCRUD.patients[i].getMedicine();
                    break;
                }
            }
        }

        private void btnclose_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
```

Individual PatientData Forms code: View MedicalReport.cs

```
public partial class ViewMedicalReportForm : Form
    {
        public ViewMedicalReportForm()
        {
            InitializeComponent();
        }

        private void ViewMedicalReportForm_Load(object sender, EventArgs e)
        {
            for (int i = 0; i < PatientCRUD.patients.Count; i++)
            {
                if (DetailForm.patientCnic == PatientCRUD.patients[i].getCnic())
                {
                    if (PatientCRUD.patients[i].PatientReport.getLaboratorytest() ==
"blood test")
                    {
                        ShowLABTest(i);
                    }
                    Show_medical_history(i);
                    Show_Diagnostic_test(i);
                }
            }
        }
        private void ShowLABTest(int i)
        {
            lblCholestrolLevel.Text =
""+PatientCRUD.patients[i].PatientReport.getCholestrolLevel();
            lblCholestrol.Text =
PatientCRUD.patients[i].PatientReport.getCholestrol();
            lbltyphoid.Text = PatientCRUD.patients[i].PatientReport.getTyphoid();
            lblmalaria.Text = PatientCRUD.patients[i].PatientReport.getMalaria();
            lblanemia.Text = PatientCRUD.patients[i].PatientReport.getAnemia();
        }
        private void Show_medical_history(int i)
        {
            lblMedicalHistory.Text = PatientCRUD.patients[i].getMedicalhistory();
        }
        private void Show_Diagnostic_test(int i)
        {
            lbldiagnostic.Text =
PatientCRUD.patients[i].PatientReport.getDiagnostictest();
            lblnormal.Text = "normal";
        }

        private void btnclose_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
```

Individual PatientData Forms code: Check DoctorSchedule.cs

```
public partial class CheckDoctorSchedule_Form : Form
    {
        DataTable doctors = new DataTable();
        public CheckDoctorSchedule_Form()
        {
            InitializeComponent();
        }

        private void CheckDoctorSchedule_Form_Load(object sender, EventArgs e)
        {

        }

        private void btnCheck_Click(object sender, EventArgs e)
        {
            doctors.Columns.Clear();
            doctors.Rows.Clear();
            DateTime date = dtdate.Value;
            string dateavailability = Convert.ToDateTime(date).Date.ToString("d");
            doctors.Columns.Add("Doctor Name", Type.GetType("System.String"));
            doctors.Columns.Add("Doctor Specialty", Type.GetType("System.String"));
            doctors.Columns.Add("Starting Time", Type.GetType("System.String"));
            doctors.Columns.Add("Ending Time", Type.GetType("System.String"));
            foreach (var doctor in Doctor_CRUD.doctors)
            {
                if (dateavailability == doctor.getScheduledate())
                {
                    doctors.Rows.Add(doctor.getDoctorname(), doctor.getProfession(),
doctor.getStartingtime(), doctor.getEndingtime());
                }
            }
            doctorGV.DataSource = doctors;
        }

        private void btnclose_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void btnCheck_MouseEnter(object sender, EventArgs e)
        {
            btnCheck.BackColor = Color.Black;
            btnCheck.ForeColor = Color.White;
        }

        private void btnCheck_MouseLeave(object sender, EventArgs e)
        {
            btnCheck.BackColor = Color.FromArgb(128, 255, 128);
            btnCheck.ForeColor = Color.Black;
        }
    }
```

Individual PatientData Forms code: Check Bill.cs

```
public partial class CheckBill_Form : Form
    {
        public CheckBill_Form()
        {
            InitializeComponent();
        }

        private void CheckBill_Form_Load(object sender, EventArgs e)
        {
            double totalpricemedicine = 0F;
            for (int i = 0; i < PatientCRUD.patients.Count; i++)
            {
                if (DetailForm.Patientname == PatientCRUD.patients[i].getName())
                {
                    ShowRoomCharges(i);
                    for (int j = 0; j < Customer_CRUD.customers.Count; j++)
                    {
                        if (PatientCRUD.patients[i].getName() ==
Customer_CRUD.customers[j].getcustomerName())
                        {
                            totalpricemedicine =
Customer_CRUD.customers[j].Calculate_Bill();
                            break;
                        }
                    }
                    Showbills(totalpricemedicine, i);
                    break;
                }
            }
        }
        private void ShowRoomCharges(int i)
        {
             lblroomcharge.Text=""+PatientCRUD.patients[i].bill.getBedcharge() *
PatientCRUD.patients[i].bill.getDaysPatientstayed();
        }
        private void Showbills(double totalpricemedicine, int i)
        {
            lblmedicinebill.Text=""+totalpricemedicine;
            lblLabtestbill.Text =""+PatientCRUD.patients[i].bill.getLabtest();
            lbltotalbill.Text=""+PatientCRUD.patients[i].bill.getTotalbill();
        }

        private void btnclose_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
```

Patient Forms code: Book Appointment.cs

```
public partial class BookAppointmentForm : Form
    {
        Appointment appointment;
        bool check = false;
        int timeSt = 0;
        int timeEnd = 0;
        public BookAppointmentForm()
        {
            InitializeComponent();
        }

        private void btncheck_Click(object sender, EventArgs e)
        {
            DateTime dateAppoint = dtDate.Value;
            string date = Convert.ToDateTime(dateAppoint).Date.ToString("d");
            string doctorname = txtdoctorName.Text;
            string name = txtname.Text;
            appointment = new Appointment(name,date,doctorname);

            for (int i = 0; i < Doctor_CRUD.doctors.Count; i++)
            {
                if (appointment.getDateappoint() ==
Doctor_CRUD.doctors[i].getScheduledate() && appointment.getDoctor() ==
Doctor_CRUD.doctors[i].getDoctorname())
                {
                    lblavailableTime.Text = "This doctor is available from: " +
Doctor_CRUD.doctors[i].getStartingtime() + " to " +
Doctor_CRUD.doctors[i].getEndingtime();
                    string startingtime_final =
Doctor_CRUD.doctors[i].getStartingtime();
                    string endingtime_final = Doctor_CRUD.doctors[i].getEndingtime();
                    timeSt = appointment.Converttime(startingtime_final);
                    timeEnd = appointment.Converttime(endingtime_final);
                    check = true;
                    break;
                }
                else
                {
                    check = false;
                }
            }
            bookAppointment_DoctorAvailable(check);
        }
        private void bookAppointment_DoctorAvailable(bool check)
        {
            if (check == false)
            {
                MessageBox.Show("This doctor is not available");
                empty();
            }
        }
        private void empty()
        {
            txtdoctorName.Text = null;
            txtname.Text = null;
            dtDate.Value = DateTime.Now;
        }
        private void Fullclean()
        {
            empty();
            lblavailableTime.Text = null;
            txttimetakeAppointment.Text = null;
        }
        private void btnbookappointment_Click(object sender, EventArgs e)
        {
            if (check == true)
            {
                string takeAppointment = txttimetakeAppointment.Text;
                appointment.setTimeappoint(takeAppointment);
                int time = appointment.Converttime(appointment.getTimeappoint());

                if (time >= timeSt && time <= timeEnd)
                {
                    MessageBox.Show("Appointment set");
                    Appointment_CRUD.add_appointmentinFile(appointment);
                    Appointment_CRUD.Add_Appointment(appointment);

Doctor_CRUD.put_DoctorAppointment_inList(Appointment_CRUD.appointments);
                }
                else
                {
                    MessageBox.Show("This doctor is not available");
                }
            }
            else { MessageBox.Show("We can't book this Appointment"); }
            this.Close();
        }

        private void btncheck_MouseEnter(object sender, EventArgs e)
        {
            btncheck.BackColor = Color.Black;
            btncheck.ForeColor = Color.White;
        }

        private void btncheck_MouseLeave(object sender, EventArgs e)
        {
            btncheck.BackColor = Color.FromArgb(128, 255, 128);
            btncheck.ForeColor = Color.Black;
        }

        private void btnbookappointment_MouseEnter(object sender, EventArgs e)
        {
            btnbookappointment.BackColor = Color.Black;
            btnbookappointment.ForeColor = Color.White;
        }

        private void btnbookappointment_MouseLeave(object sender, EventArgs e)
        {
            btnbookappointment.BackColor = Color.FromArgb(128, 255, 128);
            btnbookappointment.ForeColor = Color.Black;
        }
    }
```

Pharmacy Forms code: Add medicine.cs

```
public partial class AddMedicine_Form : Form
    {
        public AddMedicine_Form()
        {
            InitializeComponent();
        }

        private void btnAdd_Click(object sender, EventArgs e)
        {
            string MedicineName = txtMedicineName.Text;
            DateTime date = dtExpiryDate.Value;
            string Expirydate = Convert.ToDateTime(date).Date.ToString("d");
            int quantity = Convert.ToInt32(Math.Round(numQuantity.Value, 0));
            double price = Convert.ToDouble(Math.Round(numPrice.Value, 0));
            int demandcheck = quantity;
            Medicine med1 = new Medicine(MedicineName, quantity, Expirydate, price,
demandcheck);
            Medicine_CRUD.Addmedicine(med1);
            Medicine_CRUD.addMedicineinFile();
            empty();
        }
        private void empty()
        {
            txtMedicineName.Text = null;
            numQuantity.Value = 0;
            numPrice.Value = 0;
            dtExpiryDate.Value = DateTime.Now;
        }
        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form newform = new Pharmacy_Menu_Form();
            newform.Show();
            this.Hide();
        }

        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }

        private void btnAdd_MouseEnter(object sender, EventArgs e)
        {
            btnAdd.BackColor = Color.Black;
            btnAdd.ForeColor = Color.White;
        }

        private void btnAdd_MouseLeave(object sender, EventArgs e)
        {
            btnAdd.BackColor = Color.FromArgb(128, 255, 128);
            btnAdd.ForeColor = Color.Black;
        }
    }
```

Pharmacy Forms code: Update Medicine.cs

```
    public partial class UpdateMedicine_Form : Form
    {
        Medicine medicine;
        string change;
        public UpdateMedicine_Form()
        {
            InitializeComponent();
        }

        private void btnCheck_Click(object sender, EventArgs e)
        {
            string medicineName = txtMedicine.Text;
            medicine = Medicine_CRUD.Updatemedicine_Check(medicineName);
            if (medicine == null)
            {
                MessageBox.Show("This medicine is not in our database");
                txtMedicine.Text = null;
            }
            else { MessageBox.Show("Medicine found"); }
        }

        private void btncontinue_Click(object sender, EventArgs e)
        {
            change = txtchange.Text;
            if (change == "medicine name" || change == "quantity" || change ==
"Quantity" || change == "expiry date" || change == "price" || change == "Prize")
            {
                lbldepend.Text = change + ":";
            }
            else { MessageBox.Show("Enter valid change"); txtchange.Text = null; }
        }
        private void empty()
        {
            txtchange.Text = null;
            txtMedicine.Text = null;
            txtupdatebox.Text = null;
        }
        private void btnUpdate_Click(object sender, EventArgs e)
        {
            if (change == "medicine name")
            {
                medicine.setMedicinename(txtupdatebox.Text);
            }
            if (change == "quantity" || change == "Quantity")
            {
                medicine.setQuantity(int.Parse(txtupdatebox.Text));
            }
            if (change == "expiry date")
            {
                medicine.setExpirydate(txtupdatebox.Text);
            }
            if (change == "price" || change == "Prize")
            {
                medicine.setPrice(double.Parse(txtupdatebox.Text));
            }
            Medicine_CRUD.saveUpdatedmedicine_data();
            empty();
        }

        private void exitToolStripMenuItem_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form newform = new Pharmacy_Menu_Form();
            newform.Show();
            this.Hide();
        }

        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }

        private void btnCheck_MouseEnter(object sender, EventArgs e)
        {
            btnCheck.BackColor = Color.Black;
            btnCheck.ForeColor = Color.White;
        }

        private void btnCheck_MouseLeave(object sender, EventArgs e)
        {
            btnCheck.BackColor = Color.FromArgb(128, 255, 128);
            btnCheck.ForeColor = Color.Black;
        }

        private void btncontinue_MouseEnter(object sender, EventArgs e)
        {
            btncontinue.BackColor = Color.Black;
            btncontinue.ForeColor = Color.White;
        }
        private void btncontinue_MouseLeave(object sender, EventArgs e)
        {
            btncontinue.BackColor = Color.FromArgb(128, 255, 128);
            btncontinue.ForeColor = Color.Black;
        }

        private void btnUpdate_MouseEnter(object sender, EventArgs e)
        {
            btnUpdate.BackColor = Color.Black;
            btnUpdate.ForeColor = Color.White;
        }

        private void btnUpdate_MouseLeave(object sender, EventArgs e)
        {
            btnUpdate.BackColor = Color.FromArgb(128, 255, 128);
            btnUpdate.ForeColor = Color.Black;
        }
    }
```

Pharmacy Forms code: View Medicine.cs

```
public partial class ViewMedicine_Form : Form
    {
        DataTable medicines = new DataTable();
        public ViewMedicine_Form()
        {
            InitializeComponent();
        }

        private void ViewMedicine_Form_Load(object sender, EventArgs e)
        {
            medicines.Columns.Add("Medicinename", Type.GetType("System.String"));
            medicines.Columns.Add("Quantity", Type.GetType("System.String"));
            medicines.Columns.Add("Expiry Date", Type.GetType("System.String"));
            medicines.Columns.Add("Price", Type.GetType("System.String"));

            for(int i = 0; i < Medicine_CRUD.meds.Count; i++)
            {
                Medicine med = Medicine_CRUD.meds[i];
                medicines.Rows.Add(med.getMedicinename(), med.getQuantity(),
med.getExpirydate(), med.getPrice());
            }
            medicinesGV.DataSource = medicines;
        }

        private void exitToolStripMenuItem_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form newform = new Pharmacy_Menu_Form();
            newform.Show();
            this.Hide();
        }

        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }

        private void txtSearchBox_TextChanged(object sender, EventArgs e)
        {
            DataView dvPatientTable = medicines.DefaultView;
            dvPatientTable.RowFilter = "Medicinename Like '%" + txtSearchBox.Text +
"%'";
        }


        private void txtSearchBox_Enter_1(object sender, EventArgs e)
        {
            if (txtSearchBox.Text == "Medicine Name")
            {
                txtSearchBox.Text = "";
                txtSearchBox.ForeColor = Color.Black;

            }
        }

        private void txtSearchBox_Leave_1(object sender, EventArgs e)
        {
            if (txtSearchBox.Text == "")
            {
                txtSearchBox.Text = "Medicine Name";
                txtSearchBox.ForeColor = Color.DimGray;
            }
        }
    }
```

Pharmacy Forms code: Purchase Medicine.cs

```
public partial class PurchaseMedicine_Form : Form
    {
        Customer customer;

        public PurchaseMedicine_Form()
        {
            InitializeComponent();
        }

        private void btnnext_Click(object sender, EventArgs e)
        {
            string textcustomer = txtCustomername.Text;
            customer = new Customer();
            customer.setcustomerName(textcustomer);
        }

        private void btnadd_Click(object sender, EventArgs e)
        {
            Medicine purchasemed = new Medicine();
            string MedicineName = txtmedicinename.Text;
            purchasemed.setMedicinename(MedicineName);

            string quantityPurchaseInString = txtquantity.Text;
            purchasemed.setQuantity(int.Parse(quantityPurchaseInString));
            purchasemed = customer.purchasemedicine(Medicine_CRUD.meds, purchasemed);

            if (purchasemed != null)
            {
                customer.AddMedicineInList(purchasemed);
                MessageBox.Show("Added");
            }
            else
            {
                MessageBox.Show("This medicine is out of stock");
            }
            txtmedicinename.Text = null;
            txtquantity.Text = null;
        }

        private void btnpurchase_Click(object sender, EventArgs e)
        {
            Customer_CRUD.AddCustomerinList(customer);
            double totalMedicine_Bill = customer.Calculate_Bill();
            lblmedicineExpense.Text = ""+totalMedicine_Bill;
            Customer_CRUD.add_purchaseMedicineinFile();
            Medicine_CRUD.SaveChange();
            MessageBox.Show("Purchased");
            empty();
        }
        private void empty()
        {
            txtCustomername.Text = null;
            txtmedicinename.Text = null;
            txtquantity.Text = null;
            lblmedicineExpense.Text = null;
        }

        private void exitToolStripMenuItem_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form newform = new Pharmacy_Menu_Form();
            newform.Show();
            this.Hide();
        }

        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }

        private void btnnext_MouseEnter(object sender, EventArgs e)
        {
            btnnext.BackColor = Color.Black;
            btnnext.ForeColor = Color.White;
        }

        private void btnnext_MouseLeave(object sender, EventArgs e)
        {
            btnnext.BackColor = Color.FromArgb(128, 255, 128);
            btnnext.ForeColor = Color.Black;
        }

        private void btnadd_MouseEnter(object sender, EventArgs e)
        {
            btnadd.BackColor = Color.Black;
            btnadd.ForeColor = Color.White;
        }
        private void btnadd_MouseLeave(object sender, EventArgs e)
        {
            btnadd.BackColor = Color.FromArgb(128, 255, 128);
            btnadd.ForeColor = Color.Black;
        }

        private void btnpurchase_MouseEnter(object sender, EventArgs e)
        {
            btnpurchase.BackColor = Color.Black;
            btnpurchase.ForeColor = Color.White;
        }

        private void btnpurchase_MouseLeave(object sender, EventArgs e)
        {
            btnpurchase.BackColor = Color.FromArgb(128, 255, 128);
            btnpurchase.ForeColor = Color.Black;
        }
    }
```

Pharmacy Forms code: Delete Medicine.cs

```
    public partial class DeleteMedicine_Form : Form
    {
        public DeleteMedicine_Form()
        {
            InitializeComponent();
        }

        private void btndelete_Click(object sender, EventArgs e)
        {
            string deleteMedicine = txtdeleteMedicine.Text;
            bool delmedicineCheck = Medicine_CRUD.deletemedicine(deleteMedicine);
            if (delmedicineCheck == true)
            {
                MessageBox.Show("Requested medicine is deleted");
                Medicine_CRUD.DataAFTERDeleteMedicine();
            }
            else
            {
                MessageBox.Show("The medicine does not exist in our database");
            }
            txtdeleteMedicine.Text = null;
        }

        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form newform = new Pharmacy_Menu_Form();
            newform.Show();
            this.Hide();
        }

        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }

        private void btndelete_MouseEnter(object sender, EventArgs e)
        {
            btndelete.BackColor = Color.Black;
            btndelete.ForeColor = Color.White;
        }

        private void btndelete_MouseLeave(object sender, EventArgs e)
        {
            btndelete.BackColor = Color.FromArgb(128, 255, 128);
            btndelete.ForeColor = Color.Black;
        }
    }
```

Pharmacy Forms code: View Demand Medicine.cs

```
public partial class ViewDemandMedicine_Form : Form
    {
        public ViewDemandMedicine_Form()
        {
            InitializeComponent();
        }

        private void exitToolStripMenuItem_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void ViewDemandMedicine_Form_Load(object sender, EventArgs e)
        {
            int check;
            string demandmed = "";
            foreach (Medicine j in Medicine_CRUD.meds)
            {
                check = j.getQuantity();
                if (check <= j.getDemandcheck() / 2)
                {
                    demandmed = demandmed + j.getMedicinename()+",";
                }

            }
            for(int j = 0; j < demandmed.Length - 1; j++)
            {
                lblDemandmed.Text = lblDemandmed.Text + demandmed[j];
            }

        }

        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form newform = new Pharmacy_Menu_Form();
            newform.Show();
            this.Hide();
        }

        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }
    }
```

Pharmacy Forms code: View Expired Medicine.cs

```
public partial class ViewExpiredMedicine_Form : Form
    {
        public ViewExpiredMedicine_Form()
        {
            InitializeComponent();
        }

        private void btnCheck_Click(object sender, EventArgs e)
        {
            string currentdate;
            int check = 0;
            bool expiredmedicine;
            string expiredMed="";
            DateTime date = dtDateExpired.Value;
            currentdate = Convert.ToDateTime(date).Date.ToString("d");
            foreach (Medicine i in Medicine_CRUD.meds)
            {
                expiredmedicine = ExpiredMedicine.isExpired(currentdate,
i.getExpirydate());
                if (expiredmedicine == true)
                {
                    check = 1;
                    string expiredmedicines1 = i.getMedicinename();
                    ExpiredMedicine expiredmedicine1 = new
ExpiredMedicine(expiredmedicines1);

                    ExpiredMedicine_CRUD.expiredmed.Add(expiredmedicine1);
                }
            }
            ExpiredMedicine_CRUD.addExpiredmedicine();
            if (check == 1)
            {
                foreach (ExpiredMedicine i in ExpiredMedicine_CRUD.expiredmed)
                {
                    expiredMed = expiredMed + i.getExpiredmedicine() + ",";
                }
                for (int j = 0; j < expiredMed.Length - 1; j++)
                {
                    lblexpiredmed.Text = lblexpiredmed.Text + expiredMed[j];
                }
            }
            else
            {
                MessageBox.Show("No medicine is expired");
            }
        }

        private void exitToolStripMenuItem_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form newform = new Pharmacy_Menu_Form();
            newform.Show();
            this.Hide();
        }

        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Environment.Exit(1);
        }

        private void btnCheck_MouseEnter(object sender, EventArgs e)
        {
            btnCheck.BackColor = Color.Black;
            btnCheck.ForeColor = Color.White;
        }

        private void btnCheck_MouseLeave(object sender, EventArgs e)
        {
            btnCheck.BackColor = Color.FromArgb(128, 255, 128);
            btnCheck.ForeColor = Color.Black;
        }
    }
```

Pharmacy Main Menu Form: Main.cs

```
    public partial class Pharmacy_Menu_Form : Form
    {
        public Pharmacy_Menu_Form()
        {
            InitializeComponent();
        }

        private void btnAddMedicine_Click(object sender, EventArgs e)
        {
            Form newform = new AddMedicine_Form();
            newform.Show();
            this.Hide();
        }

        private void btnViewMedicine_Click(object sender, EventArgs e)
        {
            Form newform = new ViewMedicine_Form();
            newform.Show();
            this.Hide();
        }

        private void btnUpdateMedicine_Click(object sender, EventArgs e)
        {
            Form newform = new UpdateMedicine_Form();
            newform.Show();
            this.Hide();
        }

        private void btnPurchaseMedicine_Click(object sender, EventArgs e)
        {
            Form newform = new PurchaseMedicine_Form();
            newform.Show();
            this.Hide();
        }

        private void btnViewExpiredMed_Click(object sender, EventArgs e)
        {
            Form newform = new ViewExpiredMedicine_Form();
            newform.Show();
            this.Hide();
        }
        private void btnViewMedDemand_Click(object sender, EventArgs e)
        {
            Form newform = new ViewDemandMedicine_Form();
            newform.Show();
            this.Hide();
        }

        private void btnDeleteMed_Click(object sender, EventArgs e)
        {
            Form newform = new DeleteMedicine_Form();
            newform.Show();
            this.Hide();
        }

        private void Pharmacy_Menu_Form_Load(object sender, EventArgs e)
        {

        }

        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form newform = new HMSApplication();
            newform.Show();
        }

        private void menuStrip2_ItemClicked(object sender,
ToolStripItemClickedEventArgs e)
        {

        }
    }
```

Admin Main Menu Form: Main.cs

```
public partial class Admin_menu_Form : Form
    {
        public Admin_menu_Form()
        {
            InitializeComponent();
        }



        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            this.Hide();
            Form newform = new HMSApplication();
            newform.Show();
        }

        private void btnAddPatient_Click(object sender, EventArgs e)
        {
            Form newForm = new Patient_Form.Add_Patient_Form();
            newForm.Show();
            this.Hide();
        }

        private void btnViewPatient_Click(object sender, EventArgs e)
        {
            Form newForm = new Patient_Form.View_Patient_form();
            newForm.Show();
            this.Hide();
        }

        private void btnUpdatePatient_Click(object sender, EventArgs e)
        {
            Form newForm = new Patient_Form.Update_Patient_Form();
            newForm.Show();
            this.Hide();
        }

        private void btnIssueBed_Click(object sender, EventArgs e)
        {
            Form newForm = new Patient_Form.Issue_Bed_form();
            newForm.Show();
            this.Hide();
        }

        private void btnSuggestMedicine_Click(object sender, EventArgs e)
        {
            Form newForm = new Patient_Form.Suggest_Medicine_Form();
            newForm.Show();
            this.Hide();
        }

        private void btnManageDoctorSchedule_Click(object sender, EventArgs e)
        {
            Form newForm = new Patient_Form.Manage_DoctorSchedule_Form();
            newForm.Show();
            this.Hide();
        }

        private void btnManageAppointment_Click(object sender, EventArgs e)
        {
            Form newForm = new Patient_Form.ManageAppointments_form();
            newForm.Show();
            this.Hide();
        }

        private void btnGenerateReport_Click(object sender, EventArgs e)
        {
            Form newForm = new Patient_Form.Generate_medical_Report_Form();
            newForm.Show();
            this.Hide();
        }

        private void btnCheckDoctorsAppointment_Click(object sender, EventArgs e)
        {
            Form newForm = new Patient_Form.Check_Doctor_Appointment_Form();
            newForm.Show();
            this.Hide();
        }

        private void btnBilling_Click(object sender, EventArgs e)
        {
            Form newForm = new Patient_Form.Billing_Form();
            newForm.Show();
            this.Hide();
        }
        private void btnDischargePatient_Click(object sender, EventArgs e)
        {
            Form newForm = new Patient_Form.Discharge_Patient_Form();
            newForm.Show();
            this.Hide();
        }

        private void Admin_menu_Form_Load(object sender, EventArgs e)
        {

        }
    }
```

Patient Main Menu Form: Main.cs

```
    public partial class Patient_menu_form : Form
    {
        public Patient_menu_form()
        {
            InitializeComponent();
        }

        private void pictureBox1_Click(object sender, EventArgs e)
        {

        }

        private void exitToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Form newform = new HMSApplication();
            newform.Show();
            this.Hide();
        }

        private void btnDetail_Click(object sender, EventArgs e)
        {
            Form newform = new DetailForm();
            newform.Show();
            this.Hide();
        }

        private void Patient_menu_form_Load(object sender, EventArgs e)
        {



        }

        private void btnViewPrescription_Click(object sender, EventArgs e)
        {
            Form newform = new ViewPrescriptionForm();
            newform.Show();
        }

        private void btnViewCommonDiseases_Click(object sender, EventArgs e)
        {
            Form newform = new ViewCommonDiseasesForm();
            newform.Show();
        }
        private void btnViewMedicalReport_Click(object sender, EventArgs e)
        {
            Form newform = new ViewMedicalReportForm();
            newform.Show();
        }

        private void btnCheckBill_Click(object sender, EventArgs e)
        {
            Form newform = new CheckBill_Form();
            newform.Show();
        }

        private void btnCheckBedAvailability_Click(object sender, EventArgs e)
        {
            Form newform = new CheckBedAvailabilityForm();
            newform.Show();
        }

        private void btnCheckDoctorSchedule_Click(object sender, EventArgs e)
        {
            Form newform = new CheckDoctorSchedule_Form();
            newform.Show();
        }

        private void btnBookAppointment_Click(object sender, EventArgs e)
        {
            Form newform = new BookAppointmentForm();
            newform.Show();
        }

        private void btnSatisfactionSurvey_Click(object sender, EventArgs e)
        {
            Form newform = new PatientSatisfactionSurveyForm();
            newform.Show();
        }

        private void menuStrip2_ItemClicked(object sender,
ToolStripItemClickedEventArgs e)
        {

        }
    }
```

Class Details:

Muser: Inside this class I have added attributes i.e. username, password and role variable and I have added the getter setter functions to access and set the user attributes.

Patient: This class has attributes name,cnic, contact no, address, medical history, and getter setter function are also included. Other functions related to the patient are also included.

Doctor: This class has attributes of Doctor name, Profession, Schedule date, ending, and starting time. It includes my appointment list.

Customer who buys medicine: This class has the attribute Customer name and purchase medicine list.

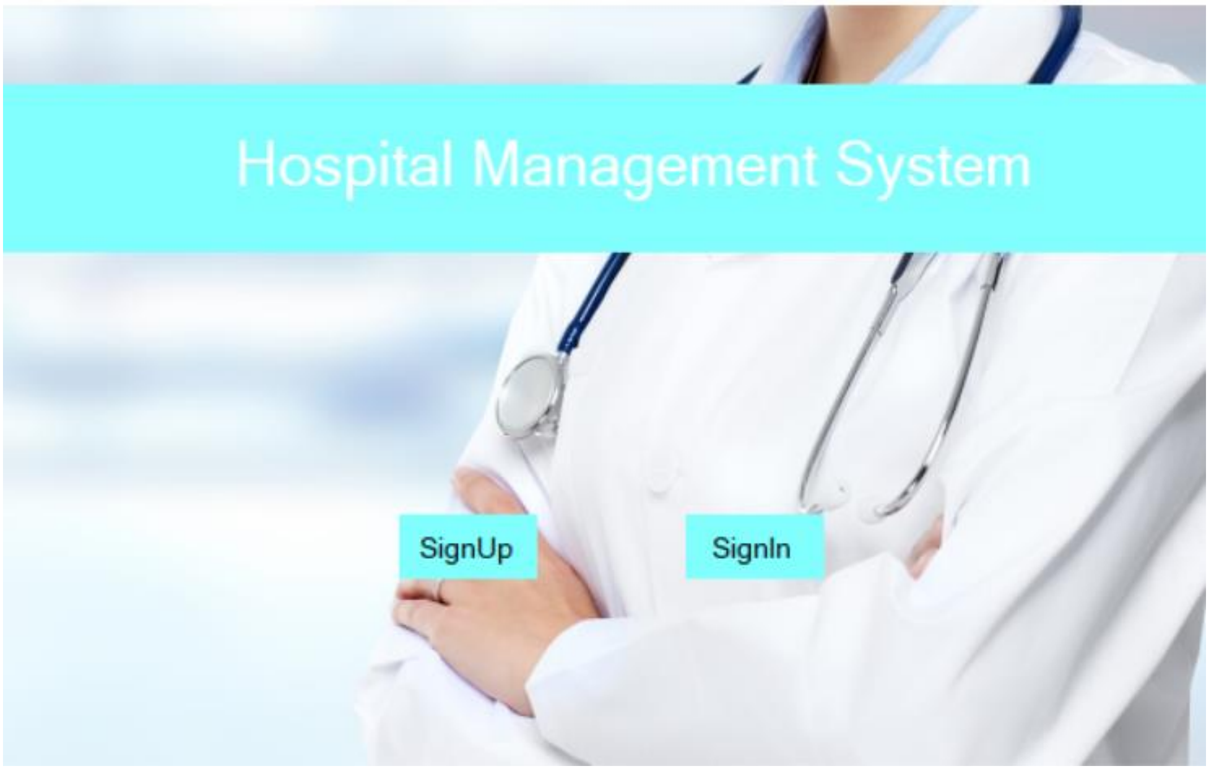
Medicine: This class has attributes of medicine name, quantity, expiry date, and price. It also includes the getter and setter functions.

DL classes: Inside DL classes I have applied all the CRUD functions in the classes to the respective list they have some kind of association and composition with the BL classes.

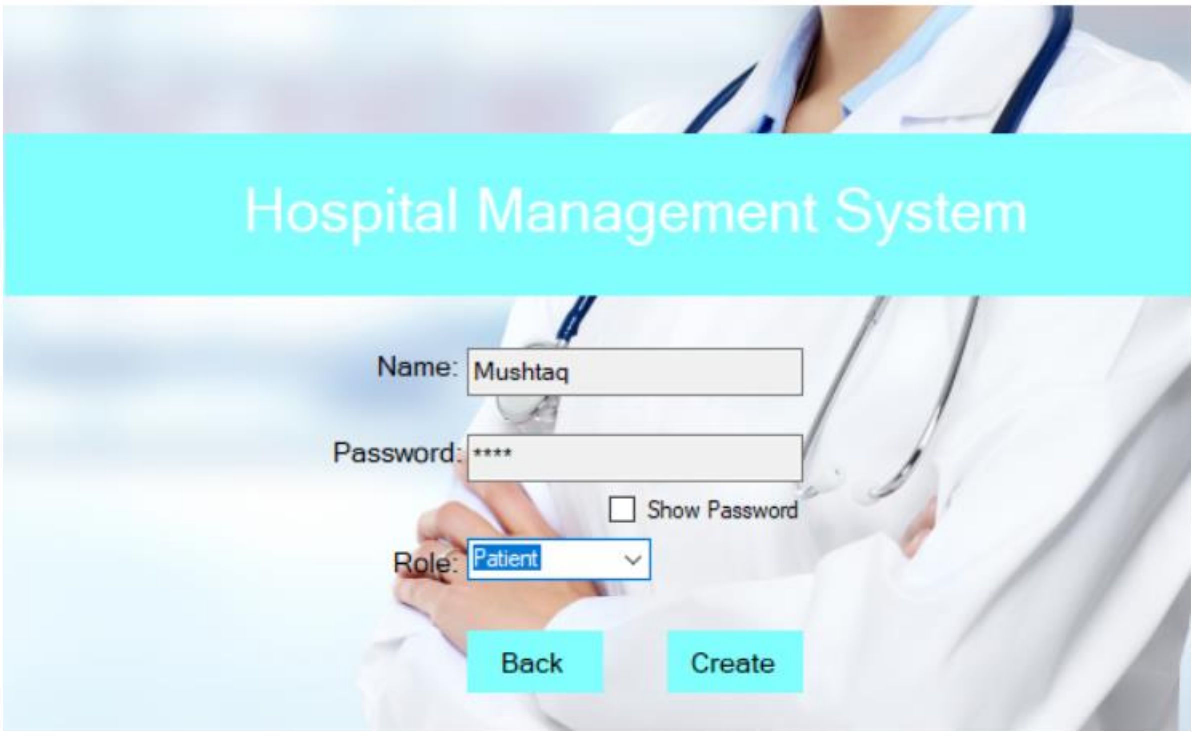
UI classes: In User UI I have all print and take input functions. It prints all the functions which may use with BL and DL classes.

WireFrames:

SignIn&SignUp Form:



SignUp Form:



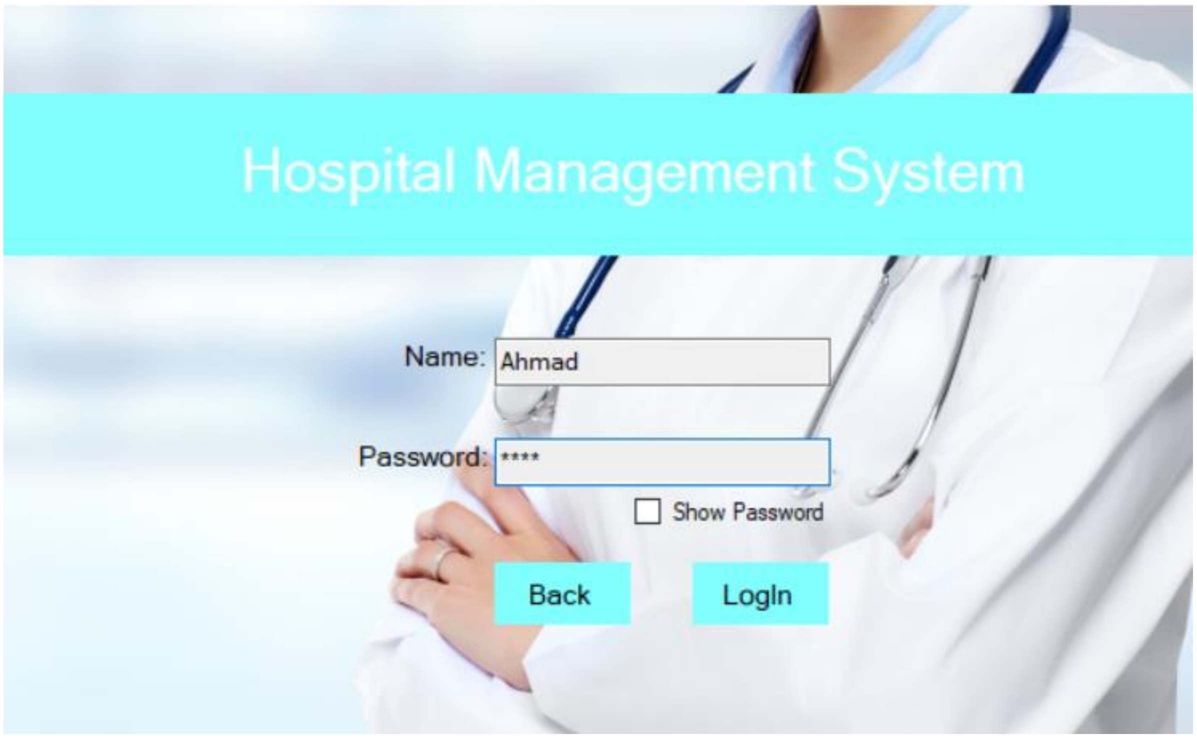
Name: Mushtaq

Password: ****

☐ Show Password

Role: Patient

SignIn Form:

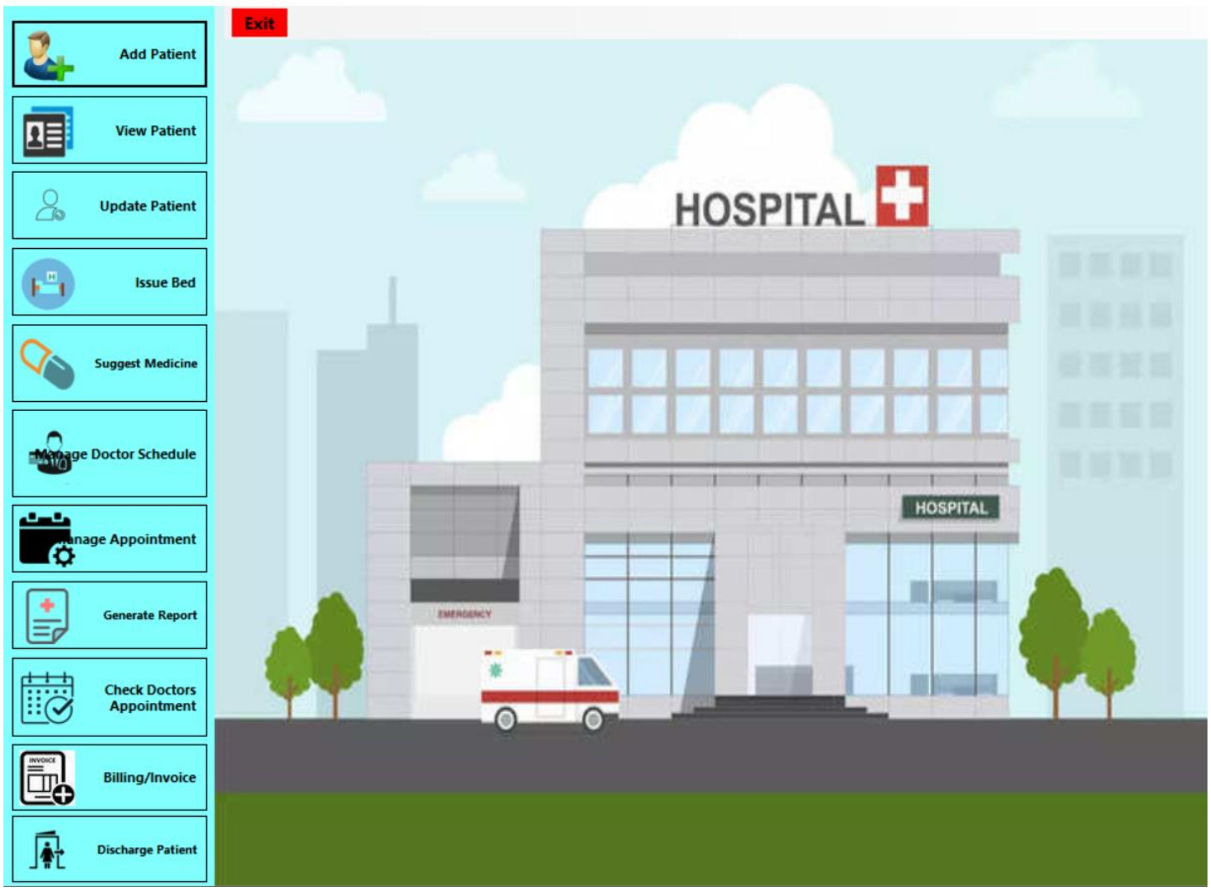


Name: Ahmad

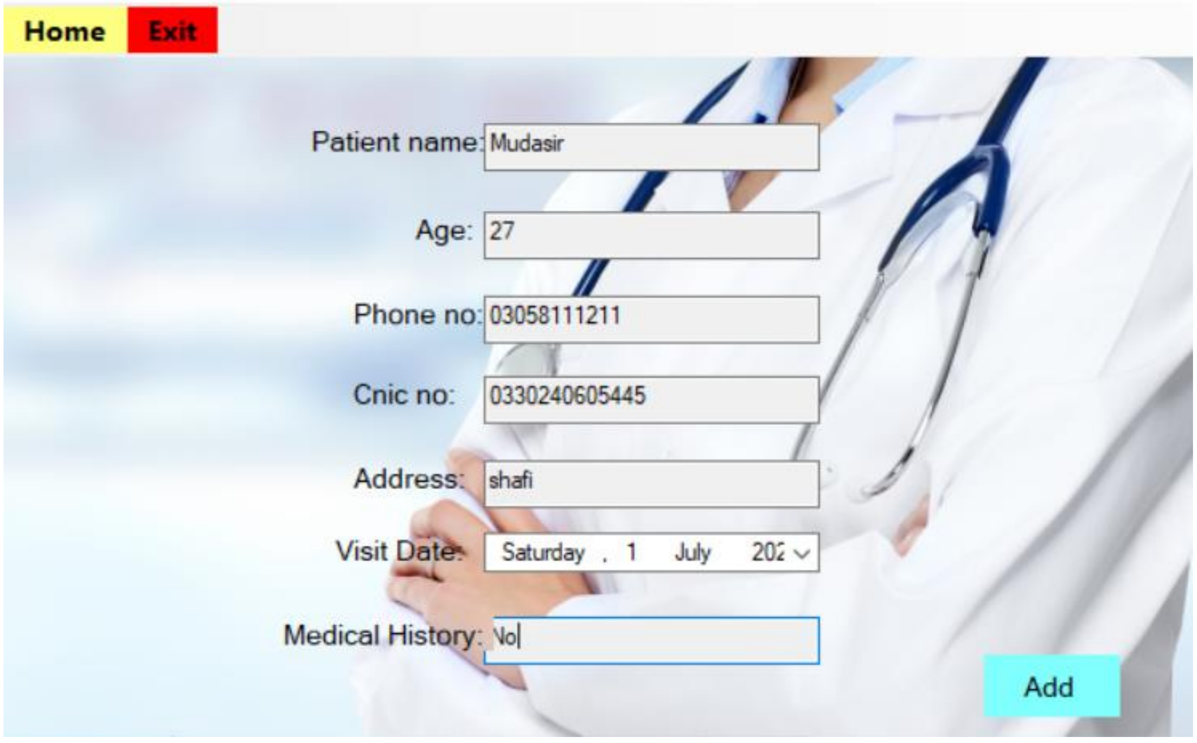
Password: ****

☐ Show Password

AdminMenu MainForm:



Add Patient Form:



Patient name: Mudasir

Age: 27

Phone no: 03058111211

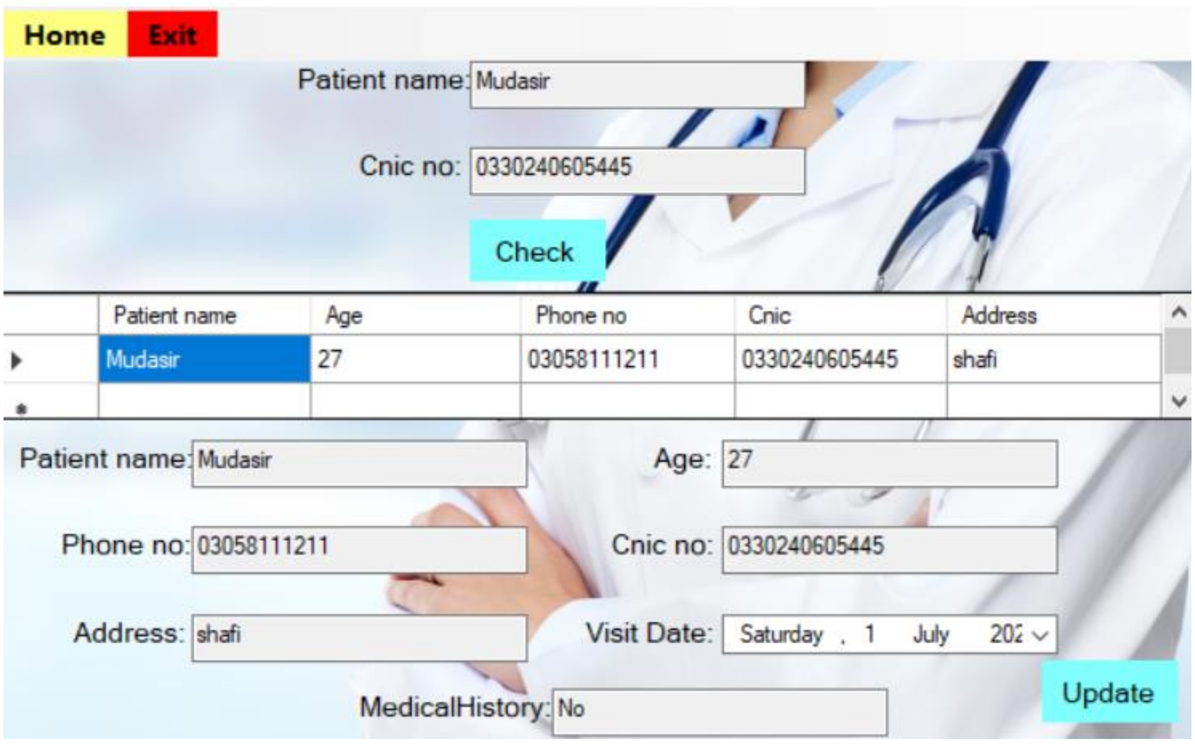
Cnic no: 0330240605445

Address: shafi

Visit Date: Saturday , 1 July 202

Medical History: No

UpdatePatient Form:



Patient name: Mudasir

Cnic no: 0330240605445

	Patient name	Age	Phone no	Cnic	Address
▶	Mudasir	27	03058111211	0330240605445	shafi
*					

Patient name: Mudasir

Age: 27

Phone no: 03058111211

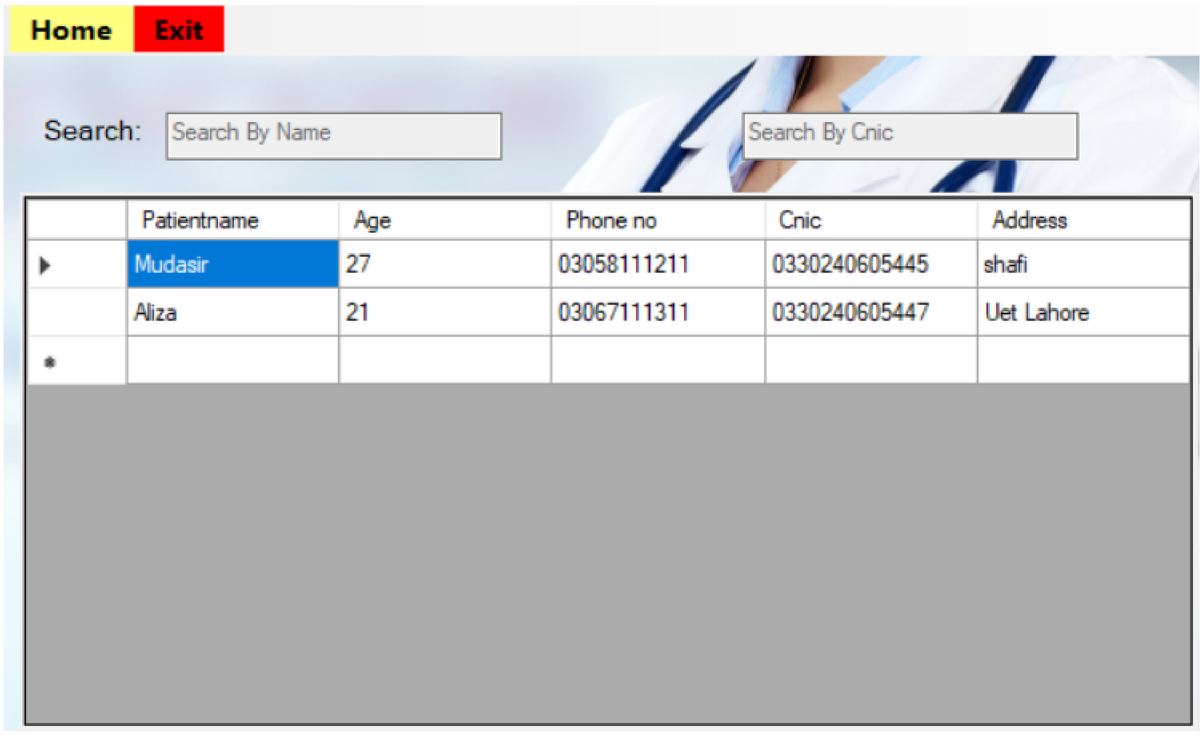
Cnic no: 0330240605445

Address: shafi

Visit Date: Saturday , 1 July 202

MedicalHistory: No

View Patient Form:

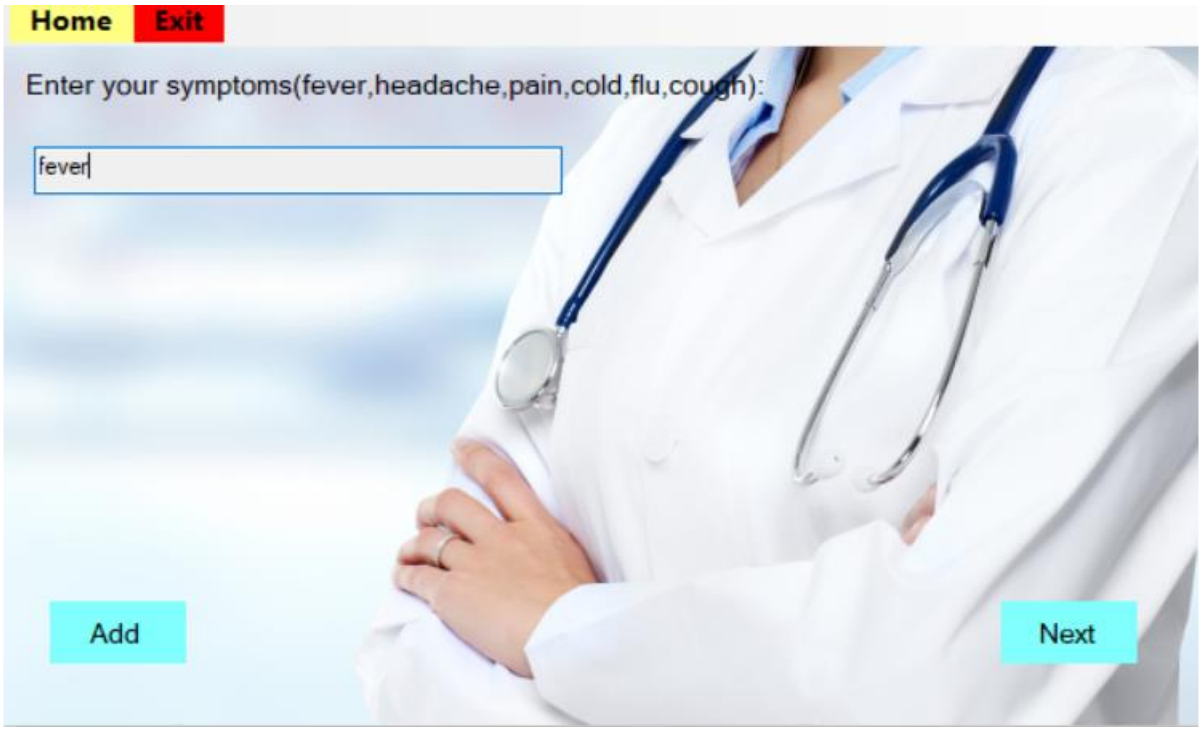


Search: Search By Name

Search By Cnic

	Patientname	Age	Phone no	Cnic	Address
▶	Mudasir	27	03058111211	0330240605445	shafi
	Aliza	21	03067111311	0330240605447	Uet Lahore
*					

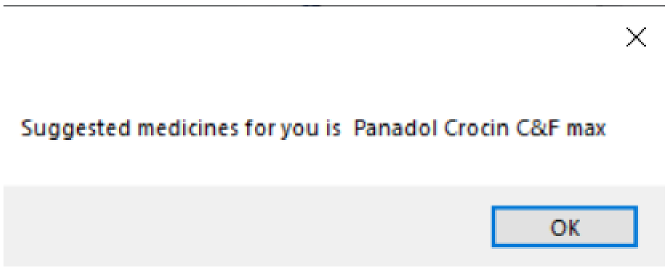
Symptoms Form:



Enter your symptoms(fever,headache,pain,cold,flu,cough):

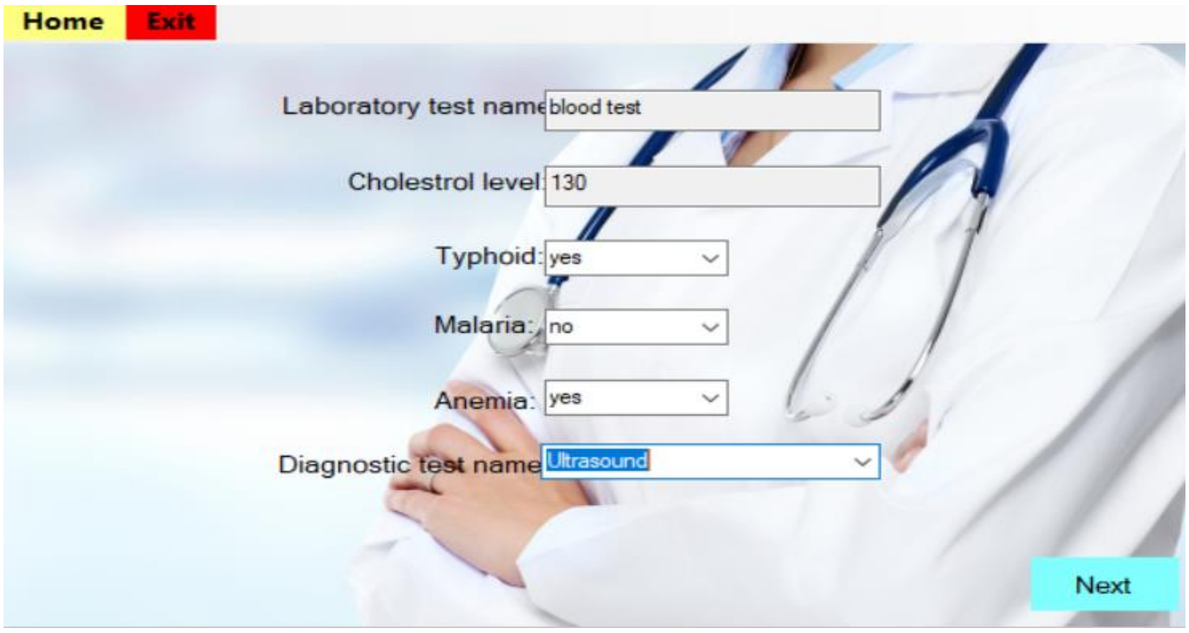
fever

Suggested Medicine According to the Symptoms:



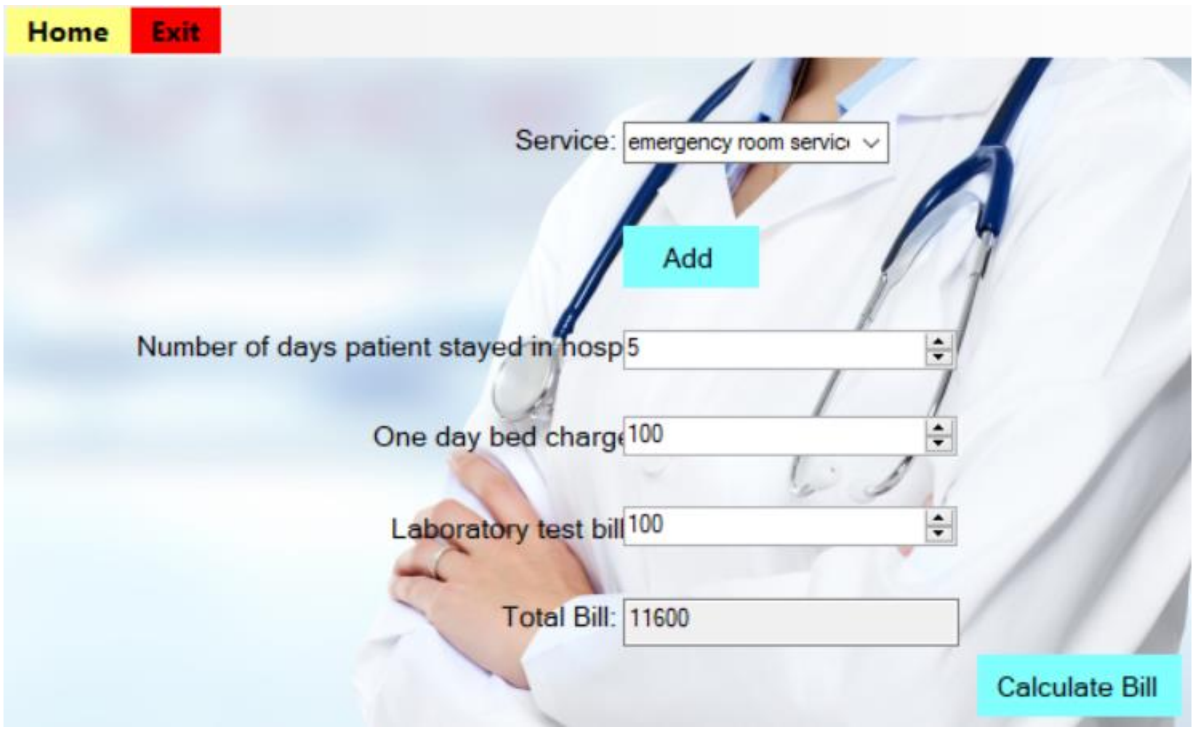
Suggested medicines for you is Panadol Crocin C&F max

Generate Report Form:



Laboratory test name	blood test
Cholestrol level:	130
Typhoid:	yes
Malaria:	no
Anemia:	yes
Diagnostic test name	Ultrasound

Billing/Invoicing Form:



Service: emergency room servic

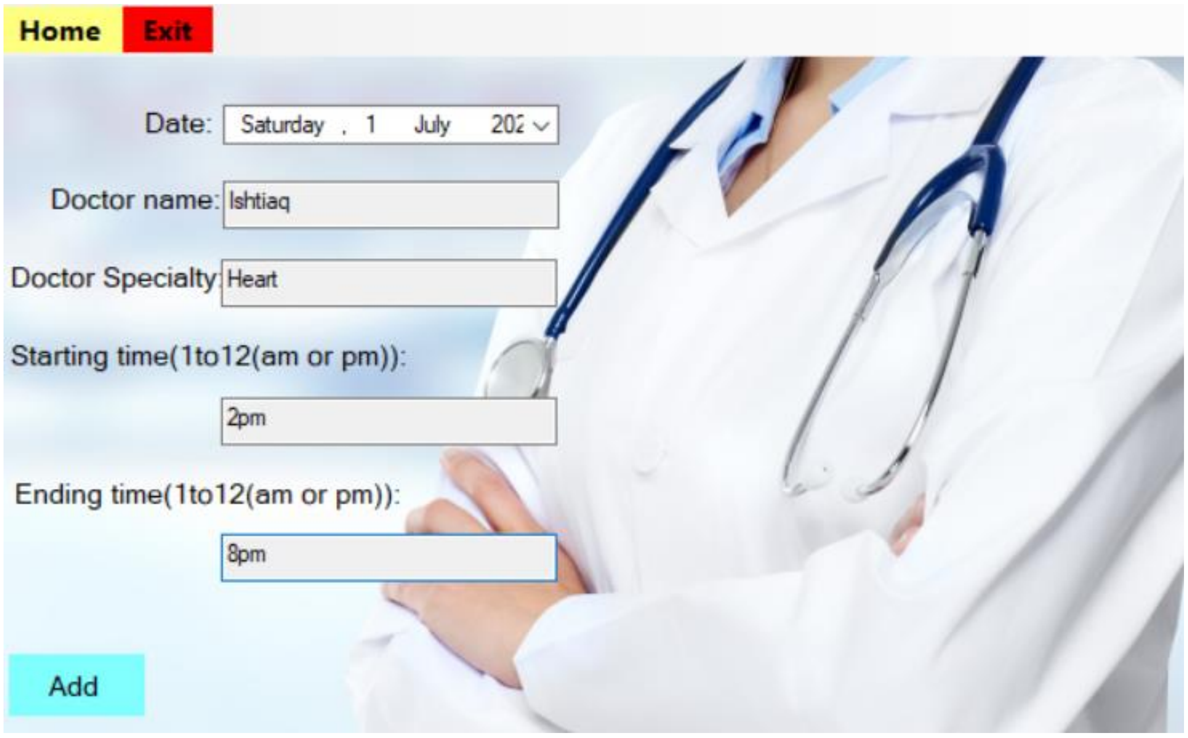
Number of days patient stayed in hosp 5

One day bed charg 100

Laboratory test bill 100

Total Bill: 11600

Set Doctor Schedule Form:



Date: Saturday , 1 July 202

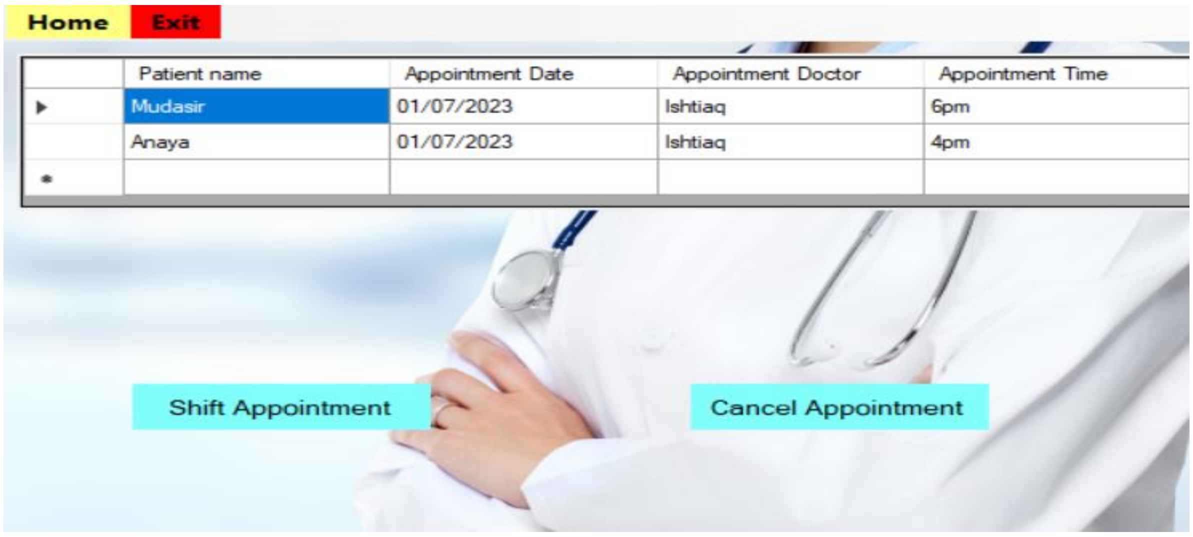
Doctor name: Ishtiaq

Doctor Specialty: Heart

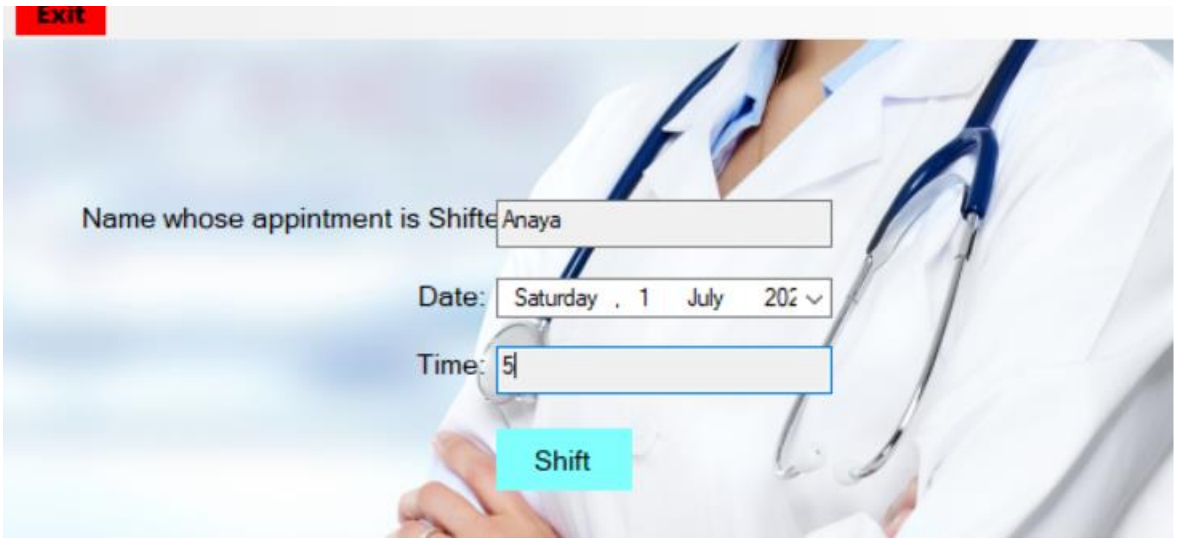
Starting time(1to12(am or pm)): 2pm

Ending time(1to12(am or pm)): 8pm

Manage Appointment Form:



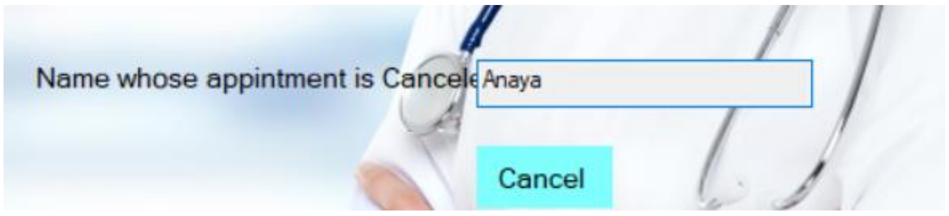
	Patient name	Appointment Date	Appointment Doctor	Appointment Time
▸	Mudasir	01/07/2023	Ishtiaq	6pm
	Anaya	01/07/2023	Ishtiaq	4pm
*				



Name whose appintment is Shifte Anaya

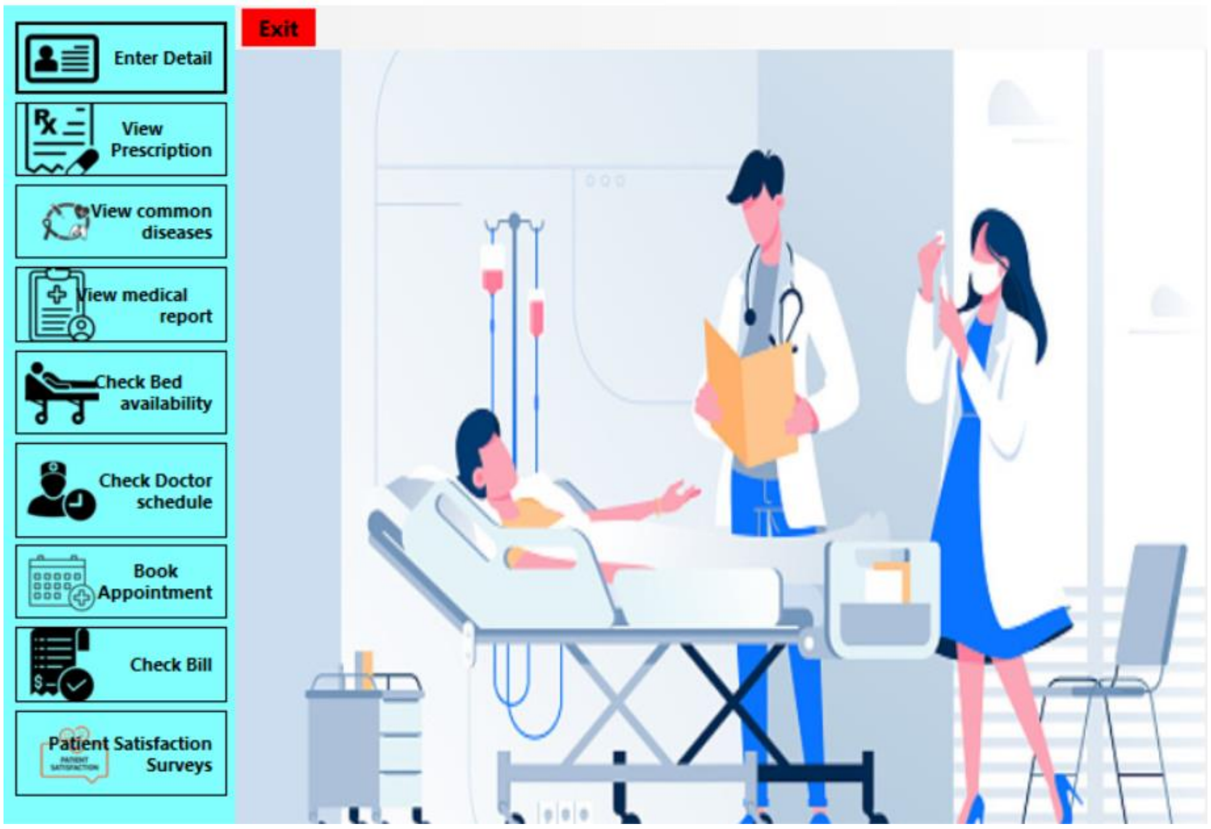
Date: Saturday , 1 July 202

Time: 5

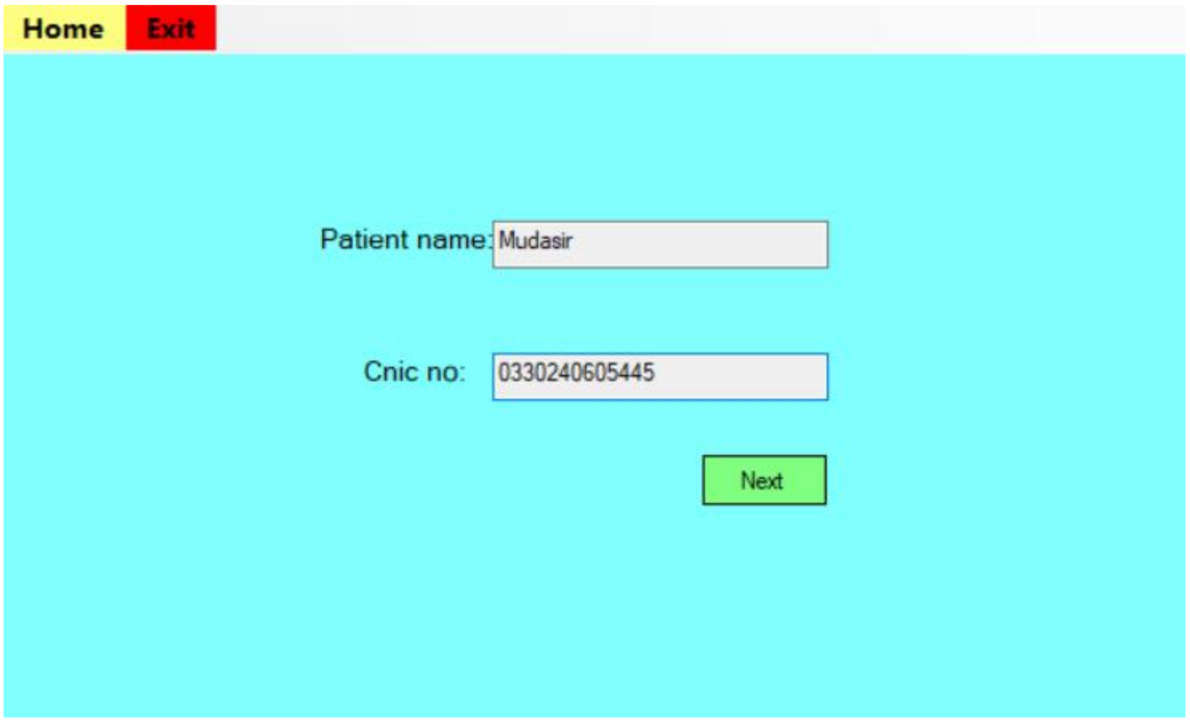


Name whose appintment is Cancel Anaya

PatientMenu MainForm:



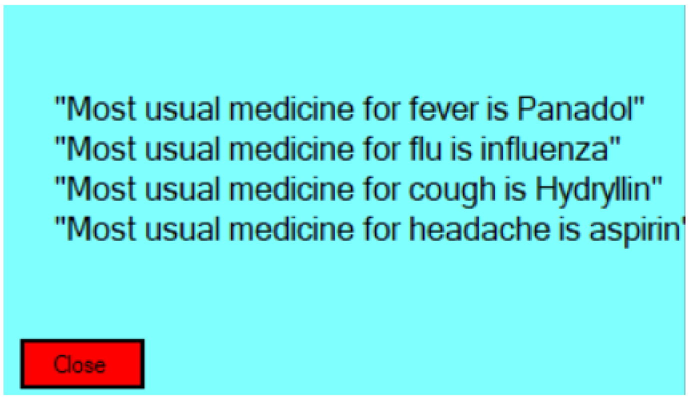
Detail Form:



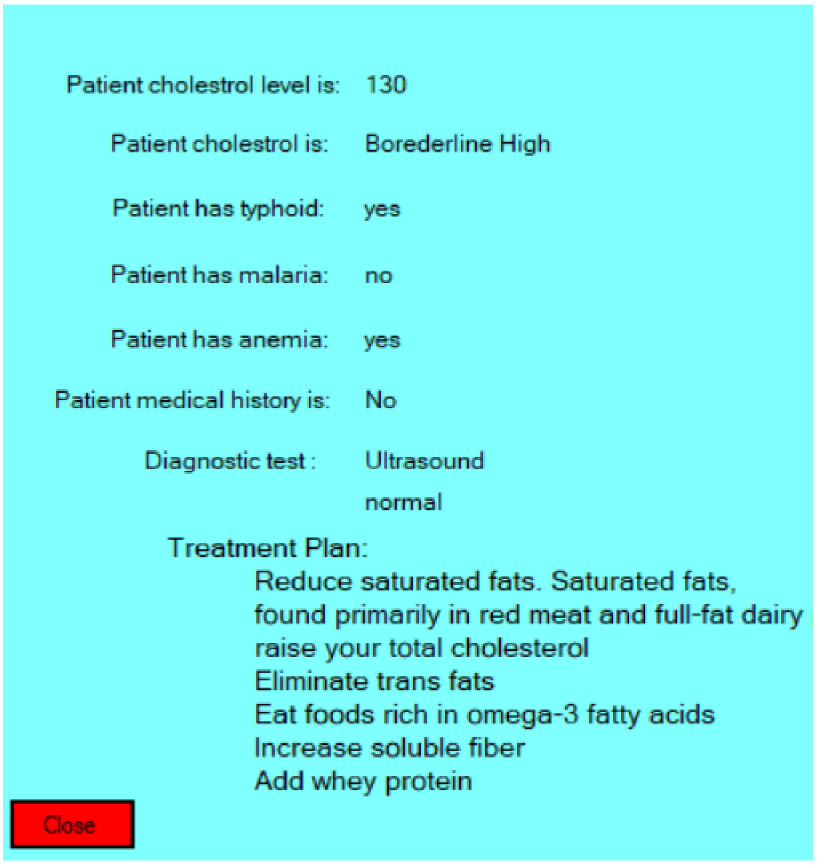
Patient name: Mudasir

Cnic no: 0330240605445

View Common Diseases:


"Most usual medicine for fever is Panadol"
"Most usual medicine for flu is influenza"
"Most usual medicine for cough is Hydryllin"
"Most usual medicine for headache is aspirin'

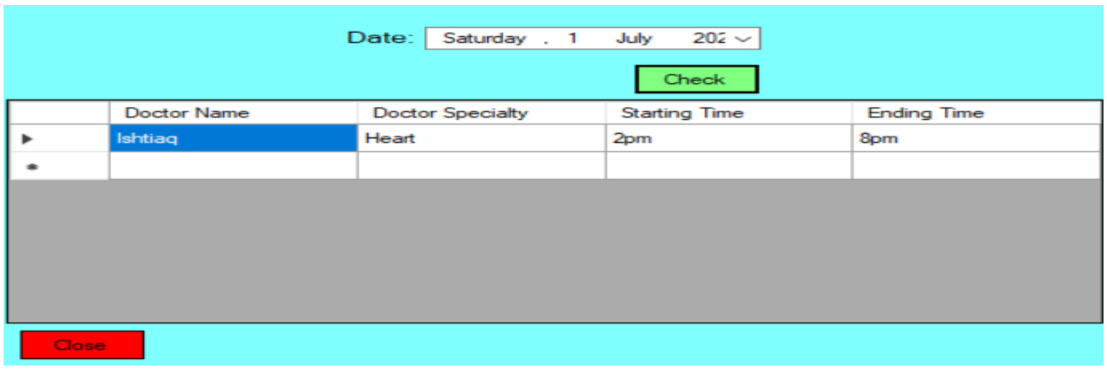
View Medical Report Form:



Patient cholestrol level is:	130
Patient cholestrol is:	Borederline High
Patient has typhoid:	yes
Patient has malaria:	no
Patient has anemia:	yes
Patient medical history is:	No
Diagnostic test :	Ultrasound
	normal

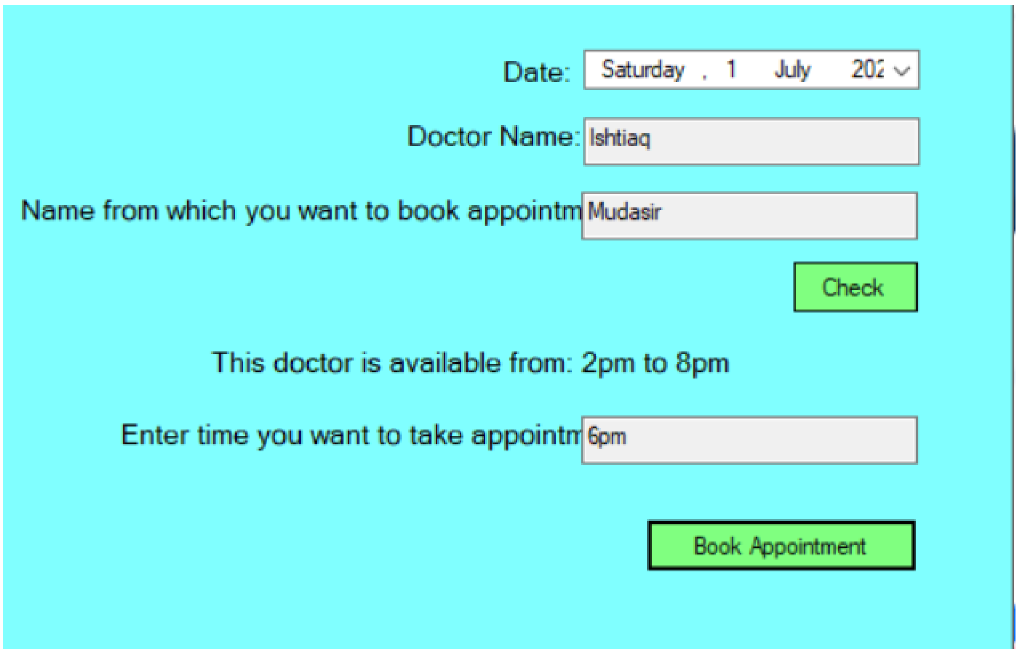
Treatment Plan:
Reduce saturated fats. Saturated fats,
found primarily in red meat and full-fat dairy
raise your total cholesterol
Eliminate trans fats
Eat foods rich in omega-3 fatty acids
Increase soluble fiber
Add whey protein

Check Doctor Schedule Form:


Date: Saturday , 1 July 202

Doctor Name	Doctor Specialty	Starting Time	Ending Time
Ishtiaq	Heart	2pm	8pm

Book Appointment Form:



Date: Saturday , 1 July 202

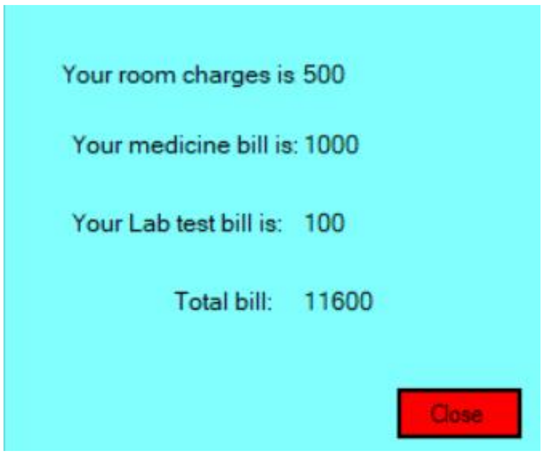
Doctor Name: Ishtiaq

Name from which you want to book appointm Mudasir

This doctor is available from: 2pm to 8pm

Enter time you want to take appointm 6pm

Check Bill:



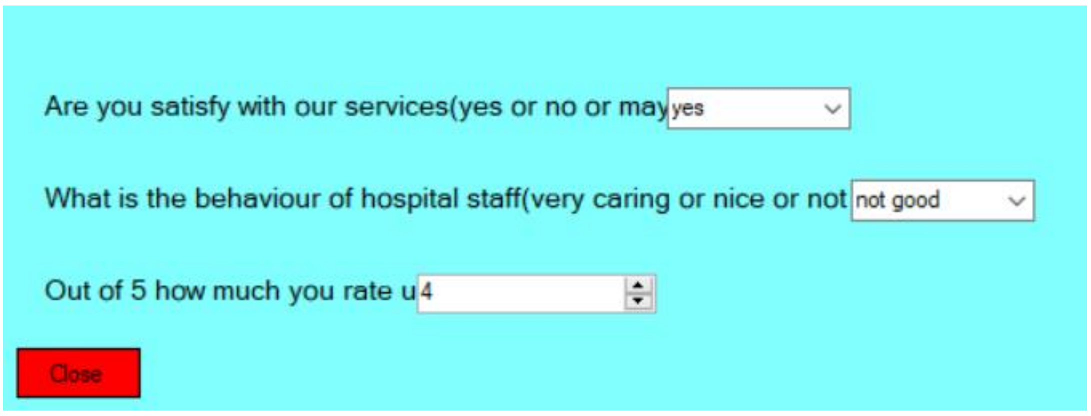
Your room charges is 500

Your medicine bill is: 1000

Your Lab test bill is: 100

Total bill: 11600

Patient Satisfaction Survey:

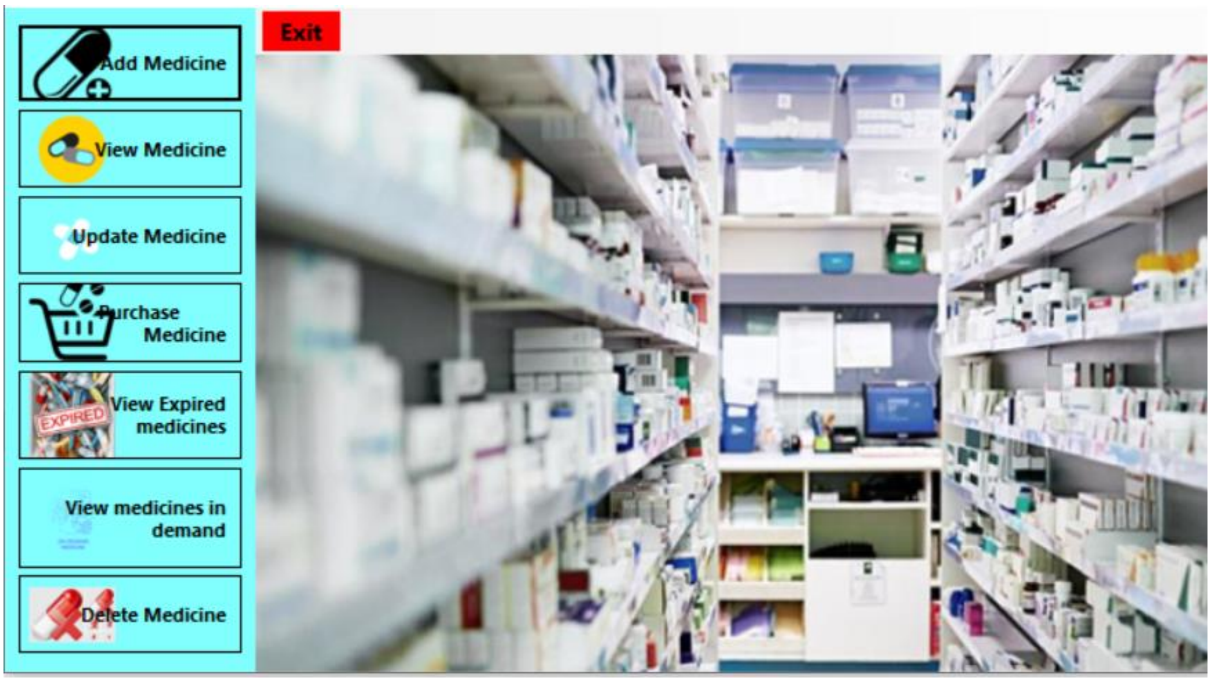


Are you satisfy with our services(yes or no or may yes

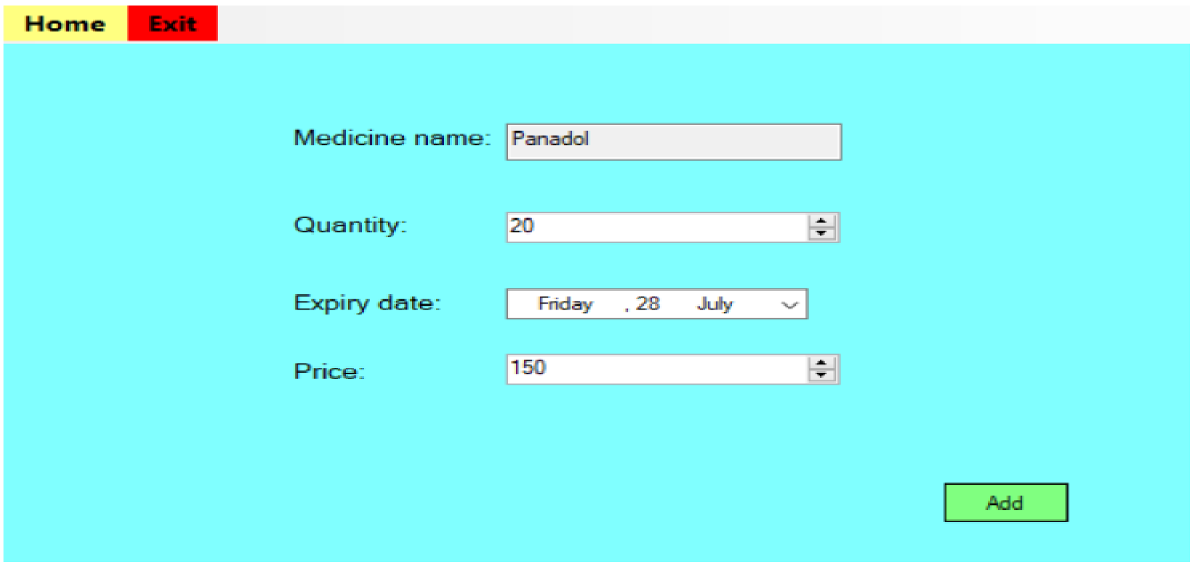
What is the behaviour of hospital staff(very caring or nice or not not good

Out of 5 how much you rate u 4

PharmacyMenu MainForm:



Add Medicine Form:



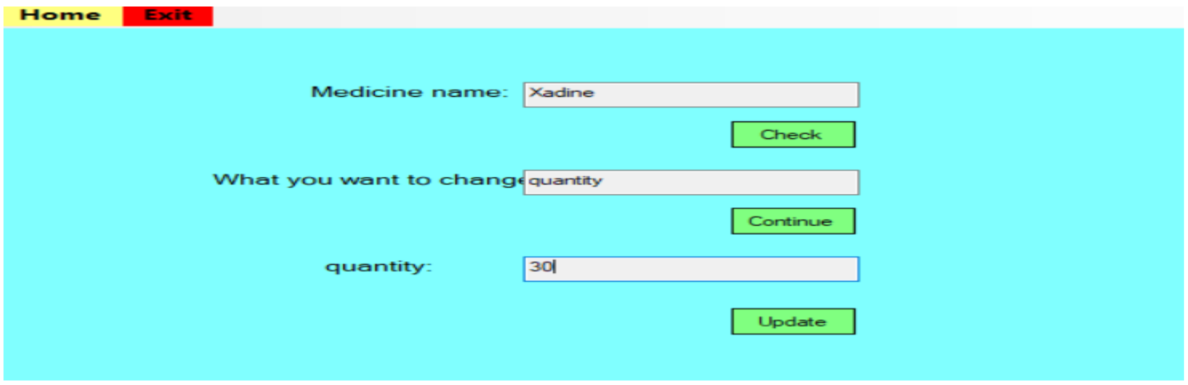
Medicine name: Panadol

Quantity: 20

Expiry date: Friday , 28 July

Price: 150

Update Medicine Form:

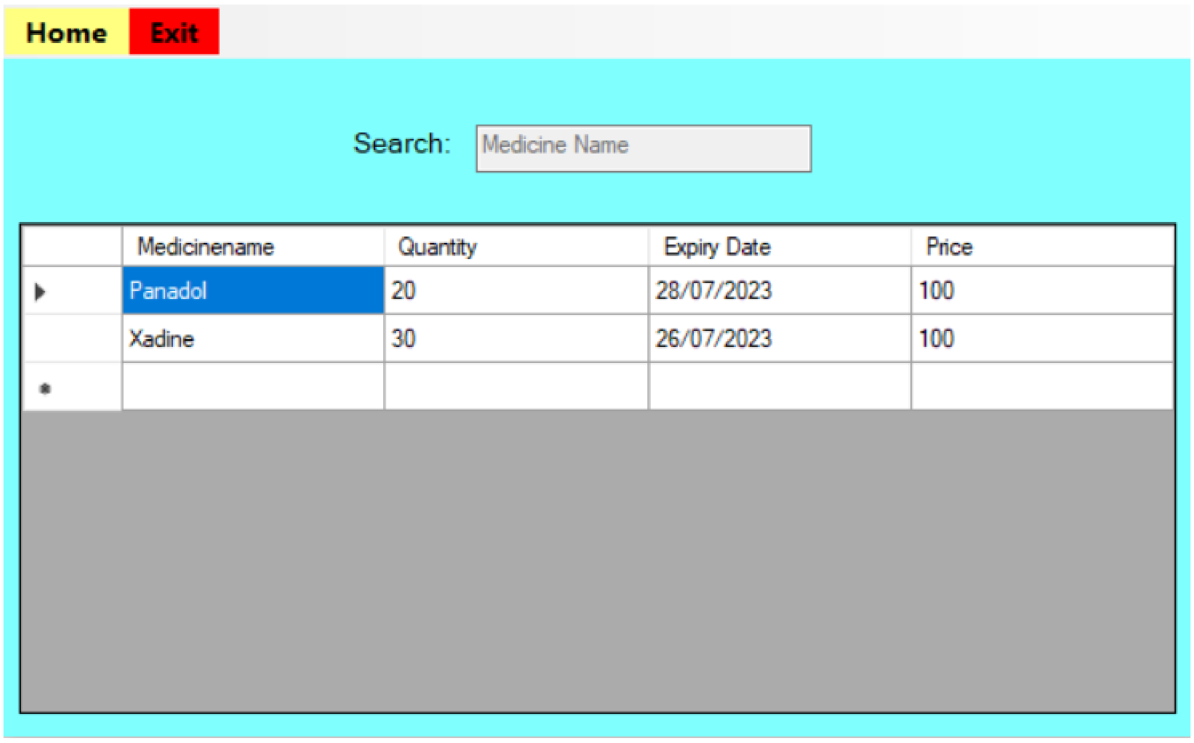


Medicine name: Xadine

What you want to chang quantity

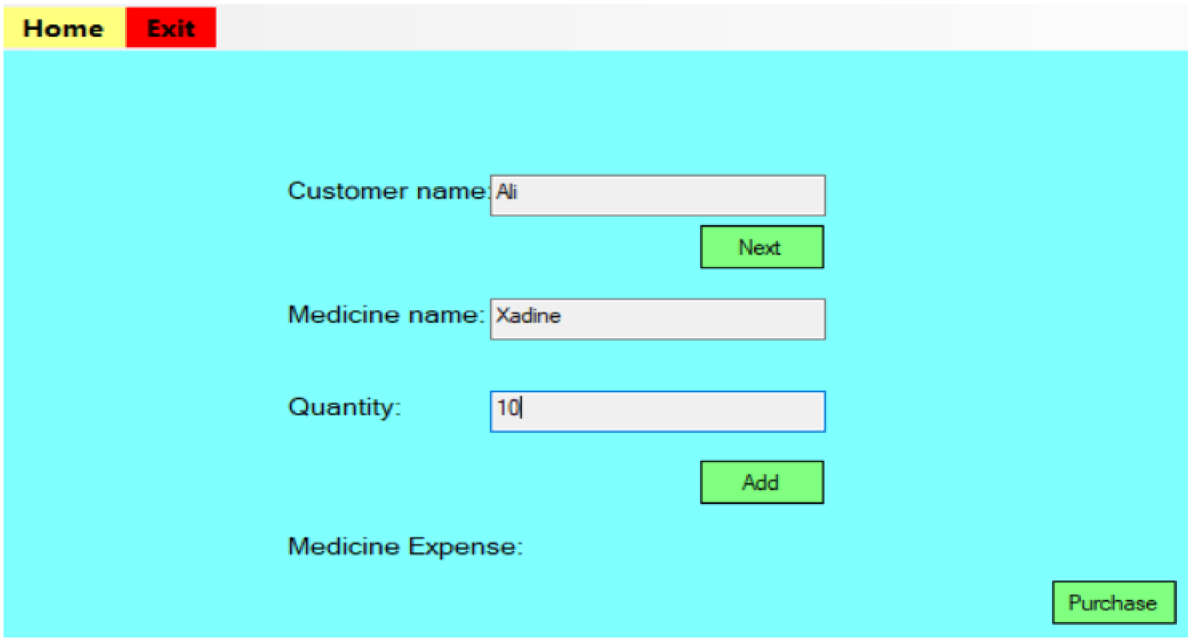
quantity: 30

View Medicine Form:



	Medicinename	Quantity	Expiry Date	Price
▶	Panadol	20	28/07/2023	100
	Xadine	30	26/07/2023	100
*				

Purchase Medicine Form:

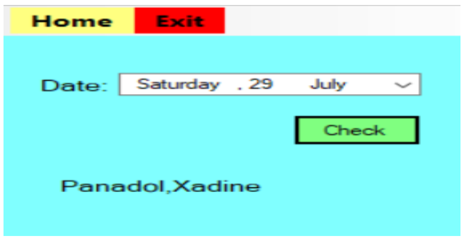


Customer name: Ali

Medicine name: Xadine

Quantity: 10

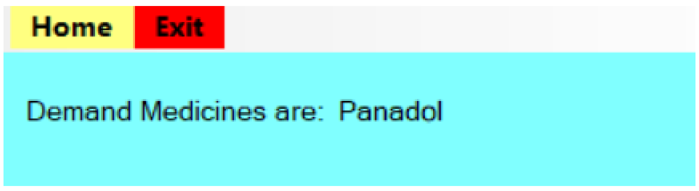
View Expired Medicine Form:



Date: Saturday , 29 July

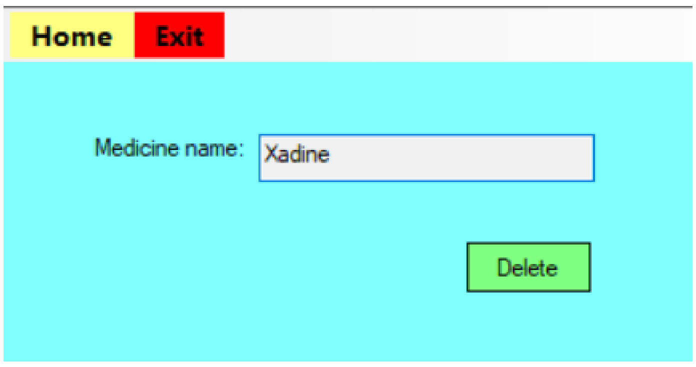
Panadol,Xadine

View Demand Medicine Form:



Demand Medicines are: Panadol

Delete Medicine Form:



Medicine name: Xadine

Conclusion:

The Hospital Management System project in C# using OOP aimed to automate various hospital operations. I faced challenges in understanding the complex domain, designing the data model, creating a user-friendly interface, implementing business logic, and ensuring system scalability. I tried to fulfill all the requirements. However, through research, learning, and problem-solving, I successfully completed the project.