```csharp
using DoctorModelLibrary;
using DoctorDALLibrary;
using DoctorBLLibrary;
using System.Diagnostics;
using System.Numerics;

namespace DoctorApp
{
    internal class Program
    {
        IDoctorService doctorservice;
        ICustomerService customerService;
        public Program()
        {
            doctorservice = new DoctorService();
            customerService = new CustomerService();
        }
        void DisplayUserTypeMenu()
        {
            Console.WriteLine("1. Register");
            Console.WriteLine("2. Login");
            //Console.WriteLine("3. Delete Product");
            //Console.WriteLine("4. Print All Products");
            Console.WriteLine("0. Exit");
        }
        void displayAdminMenu()
        {
            Console.WriteLine(" 1.Add Doctor");
            Console.WriteLine(" 2.Modify Doctor Phone");
            Console.WriteLine(" 3.Modify Doctor Experience");
            Console.WriteLine(" 4.Delete Doctor");
            Console.WriteLine(" 5.Print All Doctors");
            Console.WriteLine(" 0. Exit ");

        }
        void StartApplication()
        {
            int choice;
            do
            {
                DisplayUserTypeMenu();
                choice = Convert.ToInt32(Console.ReadLine());
                switch (choice)
                {
                    case 0:
                        Console.WriteLine("Closing....");
                        break;
                    case 1:
                        RegisterCustomer();
                        break;
                    case 2:
                        Login();
                        break;
                    default:
                        Console.WriteLine("Invalid choice. Try again");
                        break;
                }
            } while (choice != 0);
        }

        private void Login()
        {
            string username, password;
            Console.WriteLine("Welcome to zomazon");
            Console.WriteLine("Please enter your email");
            Customer customer = new Customer();
            username = Console.ReadLine();
            Console.WriteLine("Please enter the password");
            password = Console.ReadLine();
            var result = customerService.Login(username, password);
            if (result)
            {
                StartAdminActivities();
            }
            else
                Console.WriteLine("Invalid credentials");
        }
        private void RegisterCustomer()
        {
            Console.WriteLine("Welcome to zomazon");
            Console.WriteLine("Please enter your email");
            Customer customer = new Customer();
            customer.Email = Console.ReadLine();
            Console.WriteLine("Please enter the password");
            customer.Password = Console.ReadLine();
            Console.WriteLine("Please enter your age");
            customer.Age = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Please enter your phone number");
            customer.Phone = Console.ReadLine();
            try
            {
                var ressult = customerService.Register(customer);
                if (ressult != null)
                    Console.WriteLine("Congrats. Registration succesfull");
            }
            catch (Exception e)
            {

                Console.WriteLine(e.Message);
            }
        }

        void StartAdminActivities()
        {
            int choice;
            do
            {
                displayAdminMenu();
                choice = Convert.ToInt32(Console.ReadLine());
                switch (choice)
                {
                    case 0:
                        Console.WriteLine("Bye bye");
                        break;
                    case 1:
                        AddDoctor();
                        break;
                    case 2:
                        UpdatePhone();
                        break;
                    case 3:
                        UpdateExperience();
                        break;
                    case 4:
                        DeleteDoctor();
                        break;
                    case 5:
                        PrintAllDoctors();
                        break;

                    default:
                        Console.WriteLine("Invalid choice. Try again");
                        break;
                }
            } while (choice != 0);
        }
        private void PrintAllDoctors()
        {
            Console.WriteLine("***********************************");
            var doctors = doctorservice.GetDoctors();
            foreach (var item in doctors)
            {
                Console.WriteLine(item);
                Console.WriteLine("-------------------------------");
            }
            Console.WriteLine("***********************************");
        }
        void AddDoctor()
        {
            try
            {
                Doctor doctor = TakeDoctorDetails();
                var result = doctorservice.AddDoctor(doctor);
                if (result != null)
                {
                    Console.WriteLine("Doctor added");
                }
            }
            catch (FormatException e)
            {
                Console.WriteLine(e.Message);

            }
            catch (NotAddedException e)
            {
                Console.WriteLine(e.Message);
            }

        }
        Doctor TakeDoctorDetails()
        {
            Doctor doctor = new Doctor();
            Console.WriteLine("Enter Doctor Name");
            doctor.Name = Console.ReadLine();
            Console.WriteLine("Enter Doctor Qualification");
            doctor.Qualification = Console.ReadLine();
            Console.WriteLine("Enter Doctor Specialization");
            doctor.Specialization = Console.ReadLine();
            Console.WriteLine("Enter Doctor Experience");
            doctor.Experience = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Enter Doctor Fee");
            doctor.Fee = Convert.ToDouble(Console.ReadLine());
            Console.WriteLine("Enter Doctor Mobile Number");
            doctor.Phone = Console.ReadLine();
            return doctor;
        }
        int GetDoctorIdFromUser()
        {
            int id;
```

```
            Console.WriteLine("Please enter the doctor id");
            id = Convert.ToInt32(Console.ReadLine());
            return id;
        }
        private void DeleteDoctor()
        {
            try
            {
                int id = GetDoctorIdFromUser();
                if (doctorservice.Delete(id) != null)
                    Console.WriteLine("Doctor deleted");
            }
            catch (NoSuchDoctorException e)
            {
                Console.WriteLine(e.Message);
            }
        }
        private void UpdatePhone()
        {
            var id = GetDoctorIdFromUser();
            Console.WriteLine("Please enter the new phone number");
            string phone = Console.ReadLine();
            Doctor doctor = new Doctor();
            doctor.Phone = phone;
            doctor.Id = id;
            try
            {
                var result = doctorservice.ModifyPhoneNumber(id, phone);
                if (result != null)
                    Console.WriteLine("Update success");
            }
            catch (NoSuchDoctorException e)
            {
                Console.WriteLine(e.Message);
            }
        }
        private void UpdateExperience()
        {
            var id = GetDoctorIdFromUser();
            Console.WriteLine("Please enter the new experience");
            int experience = Convert.ToInt32(Console.ReadLine());
            Doctor doctor = new Doctor();
            doctor.Experience = experience;
            doctor.Id = id;
            try
            {
                var result = doctorservice.ModifyExperience(id, experience);
                if (result != null)
                    Console.WriteLine("Update success");
            }
            catch (NoSuchDoctorException e)
            {
                Console.WriteLine(e.Message);
            }
        }
        static void Main(string[] args)
        {
            Program program = new Program();
            program.StartApplication();
            //program.StartAdminActivities();
            //Console.WriteLine("Hello, World!");
        }
    }
```

```
}
```

---

```
using DoctorModelLibrary;
namespace DoctorDALLibrary
{

    public class DoctorRepository : IRepository<int, Doctor>
    {
        Dictionary<int, Doctor> doctors = new Dictionary<int, Doctor>();
        /// <summary>
        ///
        /// </summary>
        /// <param name="doctor">Doctor object that has to be added</param>
        /// <returns></returns>
        public Doctor Add(Doctor doctor)
        {
            int id = GetTheNextId();
            try
            {
                doctor.Id = id;
                doctors.Add(doctor.Id, doctor);
                return doctor;
            }
            catch (ArgumentException e)
            {
                Console.WriteLine("The doctor Id already exists");
                Console.WriteLine(e.Message);
            }
            return null;
        }
        /// <summary>
        ///
        ///
        /// </summary>
        /// <returns></returns>
        private int GetTheNextId()
        {
            if (doctors.Count == 0)
                return 1;
            int id = doctors.Keys.Max();
            return ++id;
        }
        /// <summary>
        ///
        /// </summary>
        /// <param name="id"></param>
        /// <returns></returns>
        public Doctor Delete(int id)
        {
            var doctor = doctors[id];
            doctors.Remove(id);
            return doctor;
        }


        public List<Doctor> GetAll()
        {
            var doctorList = doctors.Values.ToList();
            return doctorList;
        }
        /// <summary>
        ///
        /// </summary>
        /// <param name="id"></param>
        /// <returns></returns>
        public Doctor GetById(int id)
        {
            if (doctors.ContainsKey(id))
                return doctors[id];
            return null;

        }
        /// <summary>
        /// 
        /// </summary>
        /// <param name="doctors"></param>
        /// <returns></returns>
        public Doctor ModifyPhone(Doctor doctor)
        {
            doctors[doctor.Id] = doctor;
            return doctors[doctor.Id];
        }
        /// <summary>
        /// 
        /// </summary>
        /// <param name="doctors"></param>
        /// <returns></returns>
        public Doctor ModifyExperience(Doctor doctor)
        {
            doctors[doctor.Id] = doctor;
            return doctors[doctor.Id];
        }


    }
}
```

---

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using DoctorModelLibrary;

namespace DoctorDALLibrary
{
    public interface IRepository<K, T> where T : class
    {
        public T Add(T item);
        public T ModifyPhone(T item);
        public T ModifyExperience(T item);

        public T Delete(K id);
        public T GetById(K id);
        public List<T> GetAll();
    }
}
```

---

```
namespace DoctorModelLibrary
{
    public class Customer
    {
        public string Email { get; set; } = string.Empty;
        public int Age { get; set; } = 0;
        private string password;
        public string Password
        {
            get
            {
                return GetMaskedPassword();
            }
            set
            {
                password = value;
            }
        }
        public string Phone { get; set; } = string.Empty;

        public Customer(string email, int age, string password, string phone)
        {
            Email = email;
            Age = age;
            Password = password;
            Phone = phone;
        }

        public Customer()
        {
        }

        public bool ComparePassword(string userPassword)
        {
            return (password == userPassword) ? true : false; ;
        }
        string GetMaskedPassword()
        {
            var len = password.Length;
            string maskedPass = password.Substring(0, 2);
            for (int i = 2; i < len; i++)
            {
                maskedPass += "*";
            }
            return maskedPass;
        }
        public override string ToString()
        {
            string maskedPass = GetMaskedPassword();
            return $"Email : {Email}\nAge : {Age}\nPhone : {Phone}\nPassword : 
{maskedPass}";
        }
    }
}
```

---

```
using DoctorDALLibrary;
using DoctorModelLibrary;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DoctorBLLibrary
{
    public class CustomerService : ICustomerService
    {
        IRepository<string, Customer> repository;
        public CustomerService()
        {
            repository = new CustomerRepository();
        }
        public bool Login(string username, string password)
        {
            var myCustomer = repository.GetById(username);
            if (myCustomer != null)
            {
                if (myCustomer.ComparePassword(password))
                    return true;
            }

            return false;
        }

        public Customer Register(Customer customer)
        {
            var result = repository.Add(customer);
            if (result != null)
            {
                return result;
            }
            throw new UnableToRegisterCustomerException();
        }
    }
}
```

---

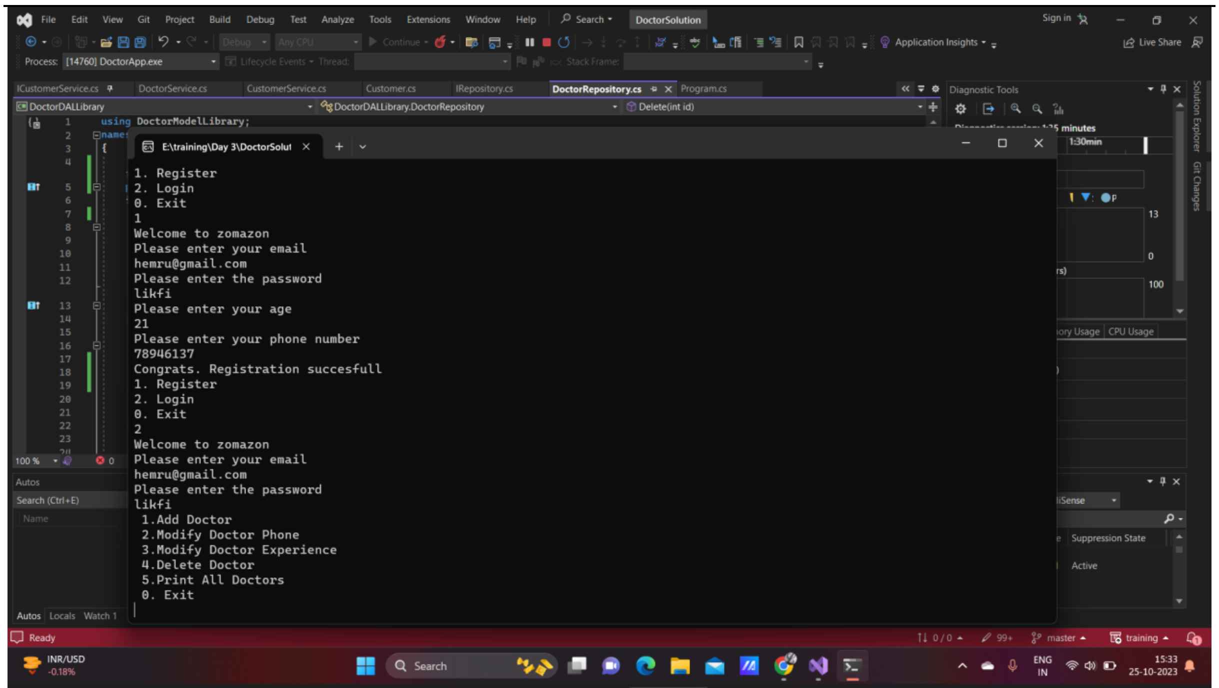
```
using DoctorDALLibrary;
using DoctorModelLibrary;
namespace DoctorBLLibrary
{
    public class DoctorService : IDoctorService
    {
        IRepository<int, Doctor> repository;
        public  DoctorService()
        {
            repository = new DoctorRepository();
        }
        public Doctor AddDoctor(Doctor doctor)
        {
            var result = repository.Add(doctor);
            if (result != null)
                return result;
            throw new NotAddedException();
        }
        public Doctor Delete(int id)
        {
            var product = GetDoctor(id);
            if (product != null)
            {
                repository.Delete(id);
                return product;
            }
            throw new NoSuchDoctorException();
        }
        public Doctor GetDoctor(int id)
        {
            var result = repository.GetById(id);
            //if (result != null)
            //    return result;
            //throw new NoSuchProductException();

            //null collasing operator
            //return result ?? throw new NoSuchProductException();

            return result == null ? throw new NoSuchDoctorException() : result;
        }
        public List<Doctor> GetDoctors()
        {
            var doctors = repository.GetAll();
            if (doctors.Count != 0)
                return doctors;
            throw new NoDoctorsAvailableException();
        }
        public Doctor ModifyPhoneNumber(int id, string Phone)
        {
            var doctor = GetDoctor(id);
            if (doctor != null)
            {
                doctor.Phone = Phone;
                var result = repository.ModifyPhone(doctor);
                return result;
            }
            throw new NoSuchDoctorException();
        }
        public Doctor ModifyExperience(int id, int Experience)
        {
            var doctor = GetDoctor(id);
            if (doctor != null)
            {
                doctor.Experience = Experience;
                var result = repository.ModifyExperience(doctor);
                return result;
            }
            throw new NoSuchDoctorException();
        }


    }
}
```

---

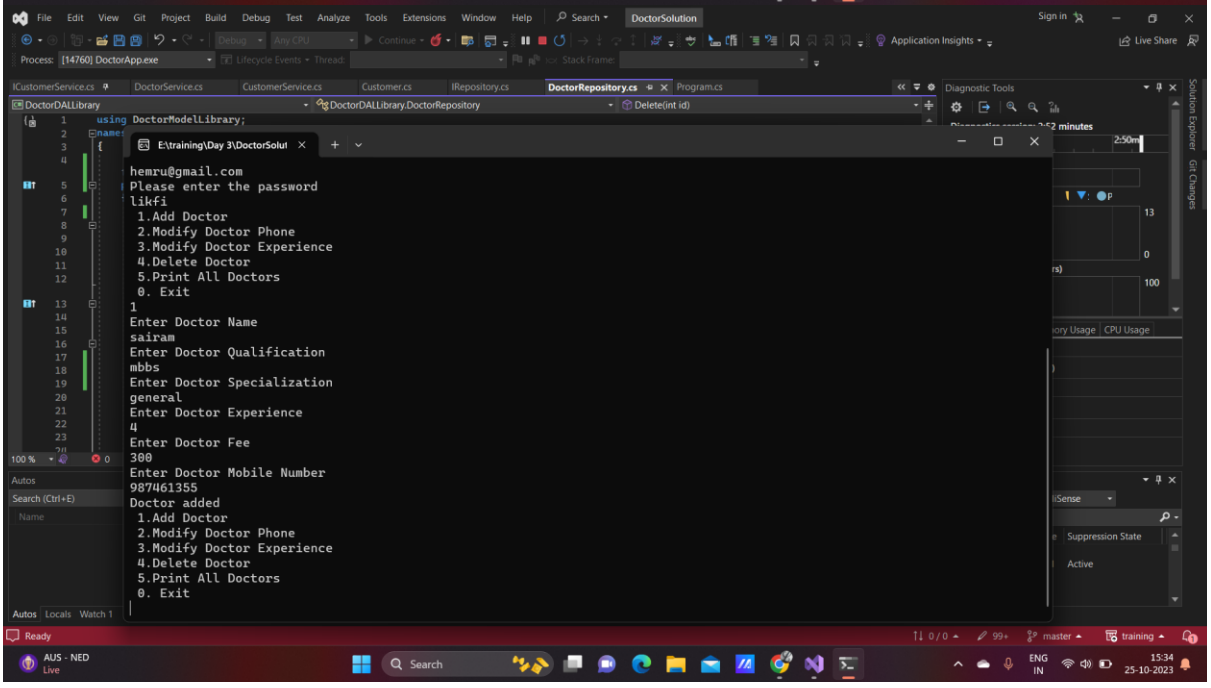
```
using DoctorModelLibrary;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DoctorBLLibrary
{
    public interface ICustomerService
    {
        public Customer Register(Customer customer);
        public bool Login(string username, string password);
    }
}
```

---

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DoctorBLLibrary
{
    public class UnableToRegisterCustomerException : Exception
    {
        string message;
        public UnableToRegisterCustomerException()
        {
            message = "Unable To Register at this moment";

        }
    }
}
```



```
1. Register
2. Login
0. Exit
1
Welcome to zomazon
Please enter your email
hemru@gmail.com
Please enter the password
likfi
Please enter your age
21
Please enter your phone number
78946137
Congrats. Registration succesfull
1. Register
2. Login
0. Exit
2
Welcome to zomazon
Please enter your email
hemru@gmail.com
Please enter the password
likfi
 1.Add Doctor
 2.Modify Doctor Phone
 3.Modify Doctor Experience
 4.Delete Doctor
 5.Print All Doctors
 0. Exit
```



```
hemru@gmail.com
Please enter the password
likfi
 1.Add Doctor
 2.Modify Doctor Phone
 3.Modify Doctor Experience
 4.Delete Doctor
 5.Print All Doctors
 0. Exit
1
Enter Doctor Name
sairam
Enter Doctor Qualification
mbbs
Enter Doctor Specialization
general
Enter Doctor Experience
4
Enter Doctor Fee
300
Enter Doctor Mobile Number
987461355
Doctor added
 1.Add Doctor
 2.Modify Doctor Phone
 3.Modify Doctor Experience
 4.Delete Doctor
 5.Print All Doctors
 0. Exit
```