**Create an application that will allow user to interact with a clinic:**

**ClinicHome.cs**

using System;

using System.Collections.Generic;

namespace ClinicApplication

{

internal class ClinicHome

{

private DoctorRepository doctorrepository;

public ClinicHome()

{

doctorrepository = new DoctorRepository();

}

private void DisplayAdminMenu()

{

Console.WriteLine("1. Add Doctor");

Console.WriteLine("2. Modify Phone");

Console.WriteLine("3. Modify Experience");

Console.WriteLine("4. Delete Doctor");

Console.WriteLine("5. Show All Doctors");

Console.WriteLine("0. Exit");

}

private void StartAdminActivities()

{

int num;

do

{

DisplayAdminMenu();

num = Convert.ToInt32(Console.ReadLine());

switch (num)

{

case 0:

Console.WriteLine("Bye bye");

break;

case 1:

doctorrepository.Add();

break;

case 2:

UpdatePhone();

break;

case 3:

UpdateExperience();

break;

case 4:

DeleteDoctor();

break;

case 5:

ShowDoctor();

break;

default:

Console.WriteLine("Invalid choice. Try again");

break;

}

}

while (num != 0);

}

private void ShowDoctor()

{

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

List<Doctor> doctors = doctorrepository.GetDoctors();

foreach (Doctor item in doctors)

{

Console.WriteLine(item);

Console.WriteLine("-------------------------------");

}

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

}

private int GetDoctorId()

{

Console.WriteLine("Please enter the doctor id");

return Convert.ToInt32(Console.ReadLine());

}

private void DeleteDoctor()

{

int doctorId = GetDoctorId();

if (doctorrepository.Delete(doctorId) != null)

{

Console.WriteLine("Doctor deleted");

}

}

private void UpdateExperience()

{

int doctorId = GetDoctorId();

Console.WriteLine("Please enter the new experience");

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

Doctor doctor = new Doctor();

doctor.Experience = experience;

doctor.Id = doctorId;

Doctor doctor2 = doctorrepository.Update(doctorId, doctor, "experience");

if (doctor2 != null)

{

Console.WriteLine("Update success");

}

}

private void UpdatePhone()

{

int doctorId = GetDoctorId();

Console.WriteLine("Please enter the new phone");

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

Doctor doctor = new Doctor();

doctor.Phone = phone;

doctor.Id = doctorId;

Doctor doctor2 = doctorrepository.Update(doctorId, doctor, "phone");

if (doctor2 != null)

{

Console.WriteLine("Update success");

}

}

private static void Main(string[] args)

{

Console.WriteLine("Welcome to my Clinic App");

ClinicHome clinicHome = new ClinicHome();

clinicHome.StartAdminActivities();

}

}

}

**Doctor.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ClinicApplication

{

internal class Doctor

{

public int Id { get; set; }

public string Name { get; set; }

public int Experience { get; set; }

public string Specialisation { get; set; }

public double Phone { get; set; }

public Doctor() {

Experience = 0;

}

public Doctor(int id, string name, int experience, string specialisation, double phone) {

Id = id;

Name = name;

Experience = experience;

Specialisation = specialisation;

Phone = phone;

}

public override string ToString()

{

return $"Doctor ID : {Id}\nName : {Name}\nSpecialisation : {Specialisation}\nExperience : {Experience} years\nPhone : {Phone}";

}

}

}

**DoctorRepository.cs:**

using System;

using System.Collections.Generic;

namespace ClinicApplication

{

internal class DoctorRepository

{

private List<Doctor> doctors;

public DoctorRepository()

{

doctors = new List<Doctor>();

}

private int GetNextId()

{

if (doctors.Count == 0)

{

return 1;

}

int id = doctors[doctors.Count - 1].Id;

return ++id;

}

private void TakeRemainingDetails(Doctor doctor)

{

Console.WriteLine("Please enter the doctor name");

doctor.Name = Console.ReadLine();

Console.WriteLine("Please enter the doctor's specialisation");

doctor.Specialisation = Console.ReadLine();

Console.WriteLine("Please enter the doctor's experience");

doctor.Experience = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Please enter the doctor's phone");

doctor.Phone = Convert.ToDouble (Console.ReadLine());

}

public Doctor Add()

{

int id = GetNextId();

Doctor doctor = new Doctor();

doctor.Id = id;

TakeRemainingDetails(doctor);

doctors.Add(doctor);

return doctor;

}

public List<Doctor> GetDoctors()

{

return doctors;

}

public Doctor GetDoctorById(int id)

{

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

{

if (doctors[i].Id == id)

{

return doctors[i];

}

}

return null;

}

public Doctor Delete(int id)

{

Doctor myDoctor = GetDoctorById(id);

if (myDoctor != null)

{

doctors.Remove(myDoctor);

Console.WriteLine("Product deleted");

return myDoctor;

}

return null;

}

public Doctor Update(int id, Doctor doctor, string choice)

{

Doctor myDoctor = GetDoctorById(id);

if (myDoctor != null)

{

if (choice == "experience")

{

if (doctor.Experience > 0)

{

myDoctor.Experience = doctor.Experience;

return myDoctor;

}

}

else if (choice == "phone")

{

if (doctor.Phone != 0)

{

myDoctor.Phone = doctor.Phone;

return myDoctor;

}

}

else

{

Console.WriteLine("Invalid choice");

}

}

Console.WriteLine("Could not update");

return null;

}

}

}

**Screenshots:**

