**Solution for Clinic App**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ClinicApp

{

internal class Clinic

{

DoctorRepository doctorRepository;

public Clinic()

{

doctorRepository = new DoctorRepository();

}

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 StartAdminActivities()

{

int choice;

do

{

displayAdminMenu();

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

switch (choice)

{

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:

PrintAllDoctors();

break;

default:

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

break;

}

} while (choice != 0);

}

private void PrintAllDoctors()

{

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

var doctors = doctorRepository.GetDoctors();

foreach (var item in doctors)

{

Console.WriteLine(item);

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

}

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

}

int GetProductIdFromUser()

{

int id;

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

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

return id;

}

private void DeleteDoctor()

{

int id = GetProductIdFromUser();

if (doctorRepository.Delete(id) != null)

Console.WriteLine("Doctor deleted");

}

private void UpdatePhone()

{

var id = GetProductIdFromUser();

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

string phone = Console.ReadLine();

Doctor doctor = new Doctor();

doctor.Phone = phone;

doctor.Id = id;

var result = doctorRepository.Update(id,doctor, "phone");

if (result!=null)

Console.WriteLine("Update success");

}

private void UpdateExperience()

{

var id = GetProductIdFromUser();

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

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

Doctor doctor = new Doctor();

doctor.Experience = experience;

doctor.Id = id;

var result = doctorRepository.Update(id, doctor, "experience");

if (result != null)

Console.WriteLine("Update success");

}

static void Main(string[] args)

{

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

Clinic clinic = new Clinic();

clinic.StartAdminActivities();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ClinicApp

{

internal class Doctor

{

public int Id { get; set; }

public string Name { get; set; }

public string Qualification { get; set; }

public string Specialization { get; set; }

public int Experience { get; set; }

public string Phone { get; set; }

public double Fee { get; set; }

public Doctor(int id, string name, string qualification, string specialization, int experience, string phone, double fee)

{

Id = id;

Name = name;

Qualification = qualification;

Specialization = specialization;

Experience = experience;

Phone = phone;

Fee = fee;

}

public Doctor()

{

}

public override string ToString()

{

return $"Id:{Id}\nName: {Name}\nQualification: {Qualification}\nSpecialization: {Specialization}\n" +

$"Experience: {Experience}\nFee: {Fee}\nPhone: {Phone}";

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ClinicApp

{

internal class DoctorRepository

{

List<Doctor> doctors;

public DoctorRepository()

{

doctors= new List<Doctor>();

}

int GetNextId()

{

if (doctors.Count == 0)

return 1;

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

return ++id;

}

void takeRemainingDoctorDetails(Doctor 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();

}

public Doctor Add()

{

int id = GetNextId();

Doctor doctor = new Doctor();

doctor.Id = id;

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

{

Doctor myDoctor = GetDoctorById(id);

if (myDoctor == null)

{

Console.WriteLine("Doctor not found");

return null;

}

if (choice == "phone")

{

if (myDoctor.Phone == "")

{

Console.WriteLine("Invalid phone value");

return null;

}

myDoctor.Phone = doctor.Phone;

return myDoctor;

}

else if (choice == "experience")

{

if (doctor.Experience != null)

{

myDoctor.Experience = doctor.Experience;

return myDoctor;

}

else

{

Console.WriteLine("Invalid experience value");

return null;

}

}

else

{

Console.WriteLine("Invalid choice");

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;

}

}

}

-------------------------------------------------------------------------------**Output:**

