Do with LINQ and without LINQ

1.Create a Class called “User” which has properties like Id, Name, Email, Location, Address and IsActive.

2.Create a New **Interface** called **IUserRepostiroy** and which has below Methods

List<User> Users();

User GetUser(int id);

List<User> DeleteUser(int id);

List<User> ActiveUsers() ;

List<User> AddUser(User user);

Implement above methods in **UserRepository** class

--------------------------------------------------------------------------------------------------------------------------------------

3.Create a new class called in that do below actions

Create a list of users object here like

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

And In constructor Add List of Users for above object with dummy data where Id is Unique for every user, make sure that this list is accessible inside class

**Example:**

**private List<Person> users= new List<Person>();**

**public PersonRepository()**

**{**

**//add dummy data here**

**}**

**public List<Person> Users()**

**{**

**return users;**

**}**

--------------------------------------------------------------------------------------------------------------------------------------

Call above Methods in Program.cs class

Program.cs

using System;

using System.Collections.Generic;

namespace practiceUserList

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Welcome!!!");

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

UserRepository userRepository = new UserRepository();

resultantList = userRepository.Users();

foreach (var item in resultantList)

{

Console.WriteLine(item.Id + "\t" + item.Name + "\t" + item.Email + "\t" + item.Address + "\t" + item.Location + "\t" + item.isActive);

}

//--------------------------------------------------------------------------------------------------------------------------------------------------------

int forNewUserInMain = userRepository.forNewUser;

forNewUserInMain++;

resultantList = userRepository.AddUser(new User() { Id = forNewUserInMain, Name = "Ram" + forNewUserInMain, Email = "email@gmail.com" + forNewUserInMain, Location = "TN", Address = "Dgl" + forNewUserInMain, isActive = true });

foreach (var item in resultantList)

{

Console.WriteLine(item.Id + "\t" + item.Name + "\t" + item.Email + "\t" + item.Address + "\t" + item.Location + "\t" + item.isActive);

}

forNewUserInMain++;

resultantList = userRepository.AddUser(new User() { Id = forNewUserInMain, Name = "Ram" + forNewUserInMain, Email = "email@gmail.com" + forNewUserInMain, Location = "TN", Address = "Dgl" + forNewUserInMain, isActive = true });

foreach (var item in resultantList)

{

Console.WriteLine(item.Id + "\t" + item.Name + "\t" + item.Email + "\t" + item.Address + "\t" + item.Location + "\t" + item.isActive);

}

//--------------------------------------------------------------------------------------------------------------------------------------------------------------------

Console.WriteLine("Enter the user id to view it's details:");

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

var getUser = userRepository.GetUser(idMainView);

if (getUser == null)

{

Console.WriteLine("Please,enter correct userId next time!!!");

}

else

{

Console.WriteLine(getUser.Id + "\t" + getUser.Name + "\t" + getUser.Email + "\t" + getUser.Address + "\t" + getUser.Location + "\t" + getUser.isActive);

}

//--------------------------------------------------------------------------------------------------------------------------------------------------------------------

Console.WriteLine("Enter the user id to delete it's details:");

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

resultantList = userRepository.DeleteUser(idMainDelete);

foreach (var item in resultantList)

{

Console.WriteLine(item.Id + "\t" + item.Name + "\t" + item.Email + "\t" + item.Address + "\t" + item.Location + "\t" + item.isActive);

}

//--------------------------------------------------------------------------------------------------------------------------------------------------------------------

resultantList = userRepository.ActiveUsers();

foreach (var item in resultantList)

{

Console.WriteLine(item.Id + "\t" + item.Name + "\t" + item.Email + "\t" + item.Address + "\t" + item.Location + "\t" + item.isActive);

}

//--------------------------------------------------------------------------------------------------------------------------------------------------------------------

Console.ReadLine();

}

}

}

User.cs

using System;

using System.Collections.Generic;

using System.Text;

namespace practiceUserList

{

public class User

{

public int Id { get; set; }

public string Name { get; set; }

public string Email { get; set; }

public string Location { get; set; }

public string Address { get; set; }

public bool isActive { get; set; }

}

}

IUserRepository.cs

using System;

using System.Collections.Generic;

using System.Text;

namespace practiceUserList

{

interface IUserRepostiroy

{

List<User> Users();

User GetUser(int id);

List<User> DeleteUser(int id);

List<User> ActiveUsers();

List<User> AddUser(User user);

}

}

UserRepository.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace practiceUserList

{

public class UserRepository : IUserRepostiroy

{

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

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

public int forNewUser;

public UserRepository()

{

for(int i=1;i<=50;i++)

{

users.Add(new User() { Id = i, Name = "Ram" + i, Email = "email@gmail.com" + i,Location = "TN",Address = "Dgl"+i,isActive=true});

forNewUser = i;

}

}

public List<User> ActiveUsers()

{

//ActiveUser = (List<User>) users.Where(a => a.isActive);

for (int i = 1; i <= users.Count; i++)

{

if (users[i].isActive)

{

ActiveUser.Add(users[i]);

}

}

return ActiveUser;

//throw new NotImplementedException();

}

public List<User> AddUser(User user)

{

forNewUser++;

users.Add(user);

return users;

//throw new NotImplementedException();

}

public List<User> DeleteUser(int id)

{

for (int i = 1; i <= users.Count; i++)

{

try

{

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

{

users.RemoveAt(i);

break;

}

}

catch

{

Console.WriteLine("User not available");

}

}

return users;

//throw new NotImplementedException();

}

public User GetUser(int id)

{

User specificUser = null;

// var specificUserLinq = users.Where(a => a.Id == id).FirstOrDefault();

for (int i = 1; i <= users.Count; i++)

{

try

{

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

{

specificUser = users[i];

break;

}

}

catch

{

Console.WriteLine("User not available");

}

}

return specificUser;

// throw new NotImplementedException();

}

public List<User> Users()

{

return users;

// throw new NotImplementedException();

}

}

}