Program.cs

using System;

using System.Collections.Generic;

using System.Text;

namespace InvestmentApp

{

public class Program

{

public static void Main(string[] args)

{

try

{

InvestmentApp();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

finally

{

}

}

public static void InvestmentApp()

{

List<InvestorDetail> l1 = new List<InvestorDetail>();

bool a = true;

while(a)

{

Console.WriteLine("1 - Add New Investor");

Console.WriteLine("2 - View Investor details");

Console.WriteLine("3 - Exit/Close the App");

int ch1 = Convert.ToInt32(Console.ReadLine());

switch (ch1)

{

case 1:

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

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

if (id == null || id == 0) throw new Exception("id cannot be empty. Please provide valid value");

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

string Name = Console.ReadLine();

if (string.IsNullOrEmpty(Name)) throw new Exception("name cannot be empty. Please provide valid value");

Console.WriteLine("Please enter the date of birth (dd/mm/yyyy)");

string dateT= Console.ReadLine();

if(string.IsNullOrEmpty(dateT)) throw new Exception("date of birth cannot be empty. Please provide valid value");

DateTime DateOfBirth= DateTime.ParseExact(dateT, "dd/mm/yyyy", null);

Console.WriteLine("Please enter the PAN card Number");

string UniqueIdentifier = Console.ReadLine();

if (string.IsNullOrEmpty(UniqueIdentifier)) throw new Exception("unique identifier cannot be empty. Please provide valid value");

Console.WriteLine("Please enter the Amount to invest");

double Amount = Convert.ToDouble(Console.ReadLine());

if (Amount <= 0) throw new Exception("Incorrect amount entered. Please provide valid value");

Console.WriteLine("Select the fund to invest");

Console.WriteLine("1 - Reliance");

Console.WriteLine("2 - Asian Paints");

Console.WriteLine("3 - Manappuram Fin");

Console.WriteLine("4 - Apollo Tyres");

Console.WriteLine("5 - Yes Bank");

int ch2 = Convert.ToInt32(Console.ReadLine());

if (ch2 == null) throw new Exception("fund name cannot be empty. Please provide valid value");

switch (ch2)

{

case 1:

InvestorDetail o1 = AddNewInvestor(id, Name, DateOfBirth, UniqueIdentifier, Amount, "Reliance");

l1.Add(o1);

Console.WriteLine("Investor details added successfully");

Console.WriteLine("Press any key to go back to the Main Menu");

Console.ReadKey();

break;

case 2:

InvestorDetail o2 = AddNewInvestor(id, Name, DateOfBirth, UniqueIdentifier, Amount, "Asian Paints");

l1.Add(o2);

Console.WriteLine("Investor details added successfully");

Console.WriteLine("Press any key to go back to the Main Menu");

Console.ReadKey();

break;

case 3:

InvestorDetail o3 = AddNewInvestor(id, Name, DateOfBirth, UniqueIdentifier, Amount, "Manappuram Fin");

l1.Add(o3);

Console.WriteLine("Investor details added successfully");

Console.WriteLine("Press any key to go back to the Main Menu");

Console.ReadKey();

break;

case 4:

InvestorDetail o4 = AddNewInvestor(id, Name, DateOfBirth, UniqueIdentifier, Amount, "Apollo Tyres");

l1.Add(o4);

Console.WriteLine("Investor details added successfully");

Console.WriteLine("Press any key to go back to the Main Menu");

Console.ReadKey();

break;

case 5:

InvestorDetail o5 = AddNewInvestor(id, Name, DateOfBirth, UniqueIdentifier, Amount, "Yes Bank");

l1.Add(o5);

Console.WriteLine("Investor details added successfully");

Console.WriteLine("Press any key to go back to the Main Menu");

Console.ReadKey();

break;

default:

Console.ReadKey();

break;

}

Console.ReadKey();

break;

case 2:

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

Console.WriteLine(" Investor Details");

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

Console.WriteLine("{0,15}{1,15}{2,15}{3,15}{4,15}", "Investor Name", "Date Of Birth", "PAN Card", "Investment Amount", "Name");

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

foreach (var ele in l1)

{

Console.WriteLine("{0,15}{1,15}{2,15}{3,15}{4,15}",ele.Name, ele.DateOfBirth.ToString("dd-mm-yyyy"), ele.UniqueIdentifier, ele.Amount, ele.FundName);

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

}

Console.WriteLine("Press any key to go back to Main Menu");

Console.ReadKey();

break;

case 3:

Console.WriteLine("Thank you");

Console.ReadKey();

a = false;

break;

default:

Console.ReadKey();

break;

}

}

}

public static InvestorDetail AddNewInvestor(int id, string name, DateTime dateOfBirth, string uniqueIdentifier, double amount, string FundName)

{

InvestorDetail obj = new InvestorDetail(id, name, dateOfBirth, uniqueIdentifier, amount, FundName);

return obj;

}

}

}

InvestorDetails.cs

using System;

using System.Collections.Generic;

using System.Text;

namespace InvestmentApp

{

public class InvestorDetail

{

public int id{get; set;}

public string Name{get; set;}

public DateTime DateOfBirth{get; set;}

public string UniqueIdentifier{get; set;}

public double Amount{get; set;}

public string FundName{get; set;}

public InvestorDetail(int id, string Name, DateTime DateOfBirth, string UniqueIdentifier, double Amount, string FundName){

this.id= id;

this.Name= Name;

this.DateOfBirth= DateOfBirth;

this.UniqueIdentifier= UniqueIdentifier;

this.Amount= Amount;

this.FundName= FundName;

}

}

}