Данчо Христов

Вътрешна система за ИТ фирма

Документация

Dancho Hristov
[COMPANY NAME]

Съдържание

Описание на приложението

„Вътрешна система за ИТ фирма“ е конзолно приложение, позволяващо създаване на записи за служители в ИТ фирма. Прототипът на системата поддържа 4 позиции и специфични за тях данни.

Потребителската информация се пази в текстов файл с конкретна стилизация. Избрах този начин за съхранение на данните защото съответства на малкия мащаб на проекта. При разширение на проекта, за съхранение на данните ще бъде използван Microsoft SQL Server.

Архитектура на проекта

Проектът се състои от базов клас (Worker), няколко наследяващи го подкласа (Ceo, Senior, Junior, Intern) и програмен файл.

Worker

```
13 references
public class Worker
{
    5 references
    public string Name { get; set; }
    5 references
    public string Position { get; set; }
    5 references
    public int YearsOld { get; set; }

    4 references
    public Worker(string name, string position, int years)
    {
        Name = name;
        Position = position;
        YearsOld = years;
    }
}
```

Ceo

```
3 references
public class Ceo : Worker
{
    2 references
    public int TaskManaged { get; set; }
    1 reference
    public Ceo(string name,string position, int years, int taskmanaged) : base(name, position , years)
    {
        TaskManaged = taskmanaged;
    }
    0 references
    public override string ToString()
    {
        string fullresult = ($"{Position},{Name},{YearsOld},{TaskManaged}");
        return fullresult;
    }
}
```

Senior

```
3 references
internal class Senior : Worker
{
    2 references
    public int ExpYears { get; set; }
    1 reference
    public Senior(string name, string position, int years, int expyears) : base(name, position, years)
    {
        ExpYears = expyears;
    }
    0 references
    public override string ToString()
    {
        string fullresult = ($"{Position},{Name},{YearsOld},{ExpYears}");
        return fullresult;
    }
}
```

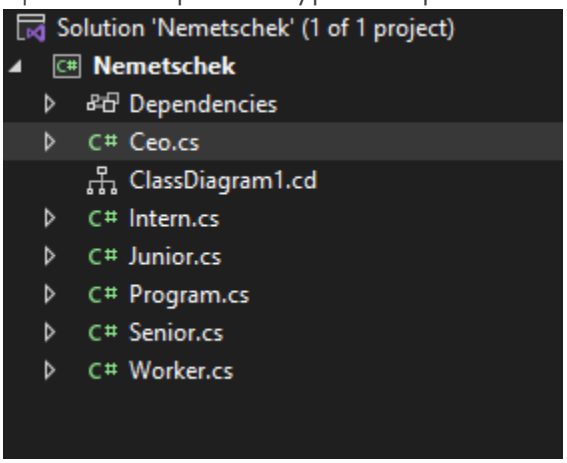
Junior

```
3 references
internal class Junior : Worker
{
    2 references
    public int TaskNum { get; set; }
    1 reference
    public Junior(string name, string position, int years, int tasknum) : base(name, position, years)
    {
        TaskNum = tasknum;
    }
    0 references
    public override string ToString()
    {
        string fullresult = ($"{Position},{Name},{YearsOld},{TaskNum}");
        return fullresult;
    }
}
```

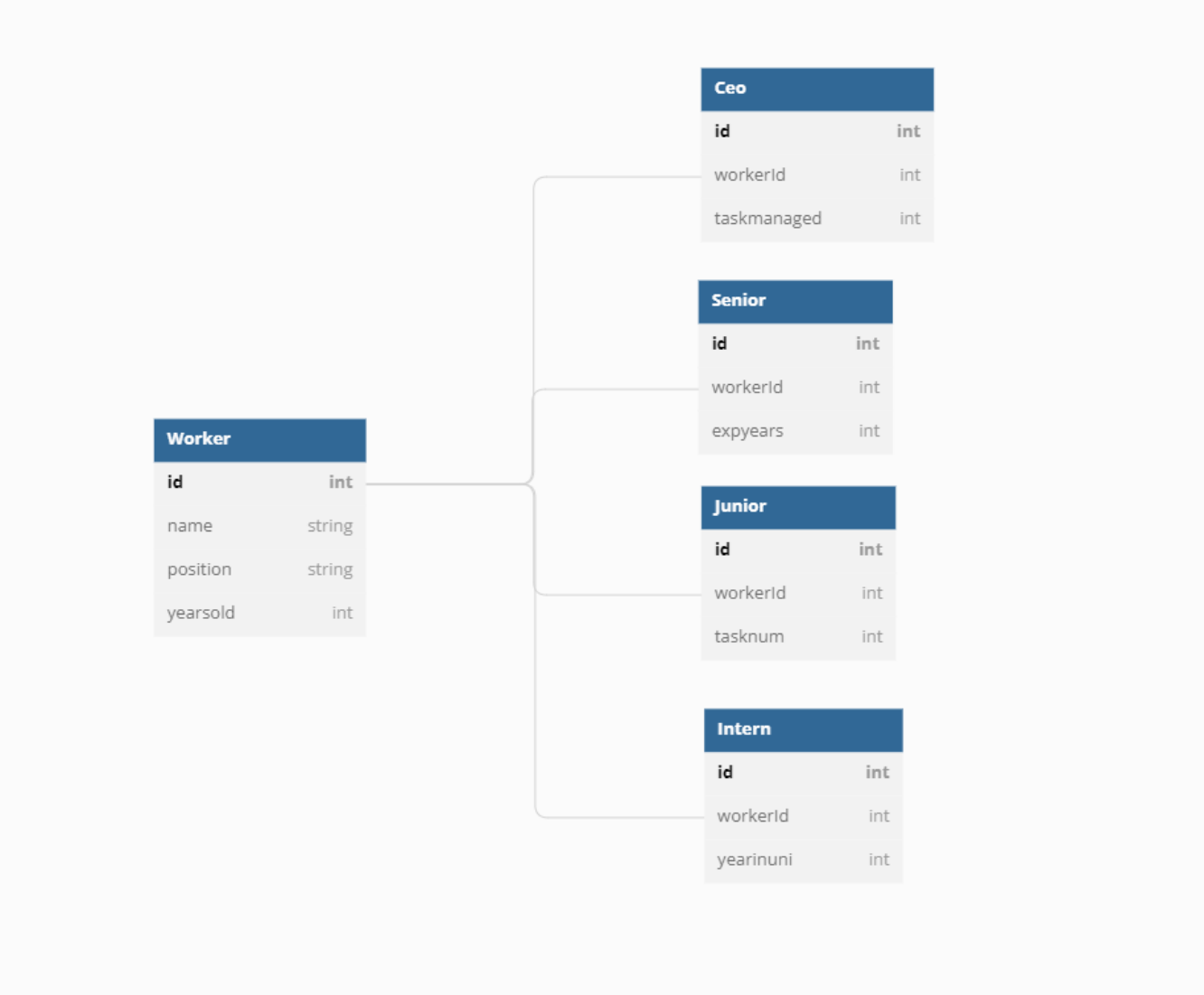
Intern

```
3 references
internal class Intern : Worker
{
    2 references
    public int YearInUni { get; set; }
    – references
    public Intern(string name, string position, int years, int yearinuni) : base(name, position, years)
    {
        YearInUni = yearinuni;
    }
    0 references
    public override string ToString()
    {
        string fullresult = ($"{Position},{Name},{YearsOld},{YearInUni}");
        return fullresult;
    }
}
```

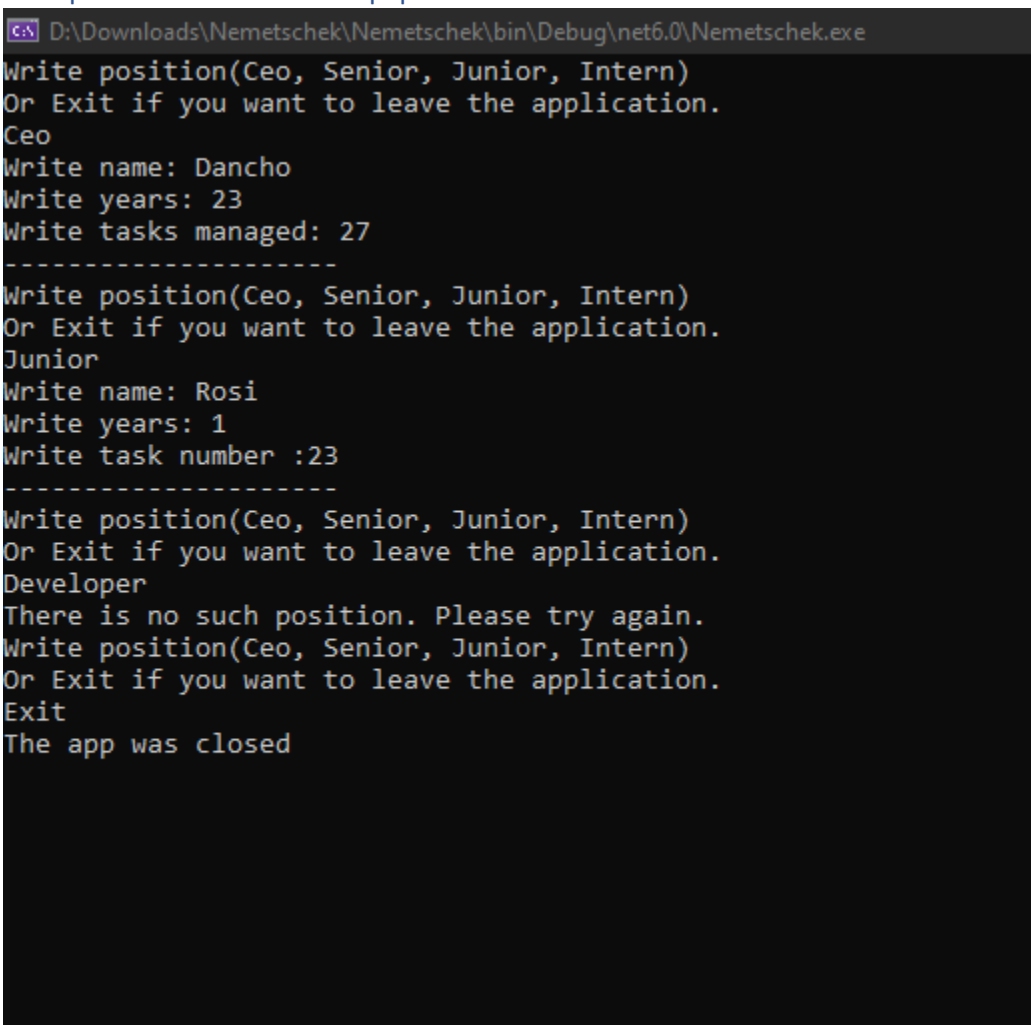
Цялостна архитектура на приложението



Модел на база данни



Потребителски интерфейс



Програмен код

```
internal class Program
    {
        static void Main(string[] args)
        {
            List<string> database =
File.ReadAllLines(@"NemetchekList.txt").ToList(); /*Reads the file data and puts it
into a list*/
            List<Worker> workers = new List<Worker>();
            while (true)
            {
                Console.Write("Write position(Ceo, Senior, Junior, Intern) \nOr Exit
if you want to leave the application. \n");
                string position = Console.ReadLine();
                if (position == "Exit")    /*The program will expect input until it
receives command "Exit"*/
                {
                    Console.WriteLine("The app was closed");
```

```
                    WriteInFile(workers);  /*After the user is done using the app
the data will be stored into the text file*/
                    break;
                }
                else if (position == "Ceo")
                {
                    Console.Write("Write name: ");
                    string name = Console.ReadLine();
                    if (!IsUserRegistered(name, position, database))    /*If the
user is not already registered it proceeds to registration*/
                    {
                        Console.Write("Write years: ");
                        int years = int.Parse(Console.ReadLine());
                        Console.Write("Write tasks managed: ");
                        int taskmanaged = int.Parse(Console.ReadLine());
                        Ceo ceo = new Ceo(name, position, years, taskmanaged);
                        workers.Add(ceo);
                    }
                    else
                    {
                        Console.WriteLine("There is already a person registered
under that name and in this position."); /*If there is already such user registered
a message will be printed*/
                    }
                    Console.WriteLine("---------------------");
                }
                else if (position == "Senior")
                {
                    Console.Write("Write name: ");
                    string name = Console.ReadLine();
                    if (!IsUserRegistered(name, position, database))
                    {
                        Console.Write("Write years: ");
                        int years = int.Parse(Console.ReadLine());
                        Console.Write("Write years if experience: ");
                        int expyears = int.Parse(Console.ReadLine());
                        Senior senior = new Senior(name, position, years, expyears);
                        workers.Add(senior);
                    }
                    else
                    {
                        Console.WriteLine("There is already a person registered
under that name and in this position.");
                    }
                    Console.WriteLine("---------------------");
                }
                else if (position == "Junior")
                {
                    Console.Write("Write name: ");
                    string name = Console.ReadLine();
                    if (!IsUserRegistered(name, position, database))
                    {
                        Console.Write("Write years: ");
                        int years = int.Parse(Console.ReadLine());
                        Console.Write("Write task number :");
                        int tasknum = int.Parse(Console.ReadLine());
                        Junior junior = new Junior(name, position, years, tasknum);
                        workers.Add(junior);
```

```
                    }
                    else
                    {
                        Console.WriteLine("There is already a person registered
under that name and in this position.");
                    }
                    Console.WriteLine("----------------------");
                }
                else if (position == "Intern")
                {
                    Console.Write("Write name: ");
                    string name = Console.ReadLine();
                    if (!IsUserRegistered(name, position, database))
                    {
                        Console.Write("Write years: ");
                        int years = int.Parse(Console.ReadLine());
                        Console.Write("Write years in university: ");
                        int yearinuni = int.Parse(Console.ReadLine());
                        Intern intern = new Intern(name, position, years,
yearinuni);
                        workers.Add(intern);
                    }
                    else
                    {
                        Console.WriteLine("There is already a person registered
under that name and in this position.");
                    }
                    Console.WriteLine("----------------------");
                }
                else
                {
                    Console.WriteLine("There is no such position. Please try
again.");
                }
            }
        }
        static bool IsUserRegistered(string name, string position, List<string>
database)
        {
            for (int i = 0; i < database.Count; i++)
            {
                List<string> data = database[i].Split(',').ToList();
                if (position == data[0] && name == data[1])
                {
                    return true;
                }
            }
            return false;
        }
        static void WriteInFile(List<Worker> workers)
        {
            List<string> resultlist = new List<string>();
            foreach (Worker worker in workers)
            {
                resultlist.Add(worker.ToString());
            }
            File.AppendAllLines(@"NemetchekList.txt", resultlist);
        }
```