# Упражнение: Връзка между C# и база данни

## Импортирайте базата данни SoftUni

Импортирайте SoftUni DB в SQL Management Studio (ако все още не е импортиран), като **изпълните** предоставения **.sql** скрипт.

****

## Генерирайте DB-First ORM модел на база данни

Моделирайте съществуващата база данни, като използвате **Database First** в Entity Framework Core.

**Създайте ново .NET Core приложение**

Първо създайте ново празно **конзолно приложение** и след като бъде създадено, отворете **Package Manager** **конзолата**:



Ще прилича на това:



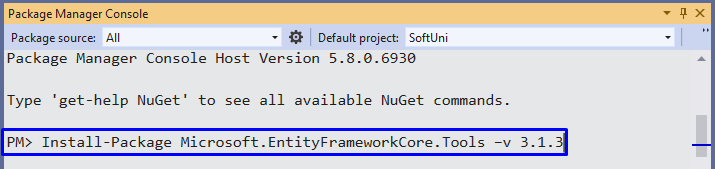
### Инсталирайте SQL Server Tools за Entity Framework

Използвайте го, за да изпълните **една по една** следните команди:

|  |
| --- |
| Install-Package Microsoft.EntityFrameworkCore.Tools -v 3.1.3 |
| Install-Package Microsoft.EntityFrameworkCore.SqlServer -v 3.1.3 |
| Install-Package Microsoft.EntityFrameworkCore.SqlServer.Design |

**Важно**: ако **Package Manager конзолата** ви даде **грешка**, докато се опитвате да **изпълните дадените команди**, опитайте да използвате следните:

|  |
| --- |
| Install-Package Microsoft.EntityFrameworkCore.Tools -Version 3.1.3 |
| Install-Package Microsoft.EntityFrameworkCore.SqlServer -Version 3.1.3 |
| Install-Package Microsoft.EntityFrameworkCore.SqlServer.Design |



Това са **пакетите**, от които ще се нуждаете, за да направите **скафоулд** на нашия **SoftUniContext** от **SoftUni** **базата данни**.

**Генерирайте EF модел на данни от съществуваща БД**

След това трябва да **изпълним** **командата** за **скафоулд** на нашия **контекстен клас**. Ще се състои от 4 неща:

* Първо, името на командата:

|  |
| --- |
| Scaffold-DbContext |

* Второ, връзката, която ще използваме (нашият connection string):

|  |
| --- |
| -Connection "Server=<ServerName>;Database=<DatabaseName>;Integrated Security=True;" |

За **ServerName** използвайте името на вашия локална инстанция на MS SQL Server или ".".

За **DatabaseName** използвайте името на базата данни, която искате да използвате, в случая – **SoftUni**.

* Трето, трябва да декларираме нашия service provider, ще използваме **Microsoft.EntityFrameworkCore.SqlServer**:

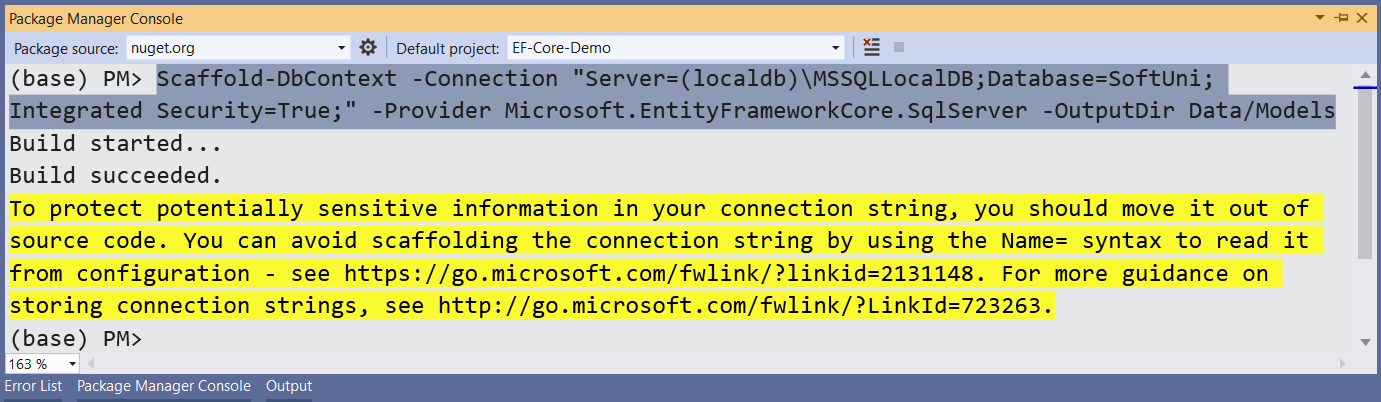
|  |
| --- |
| -Provider Microsoft.EntityFrameworkCore.SqlServer |

* И четвъртото нещо, което ще направим, е да му дадем директория, където ще отидат всички наши модели (напр. **Models**):

|  |
| --- |
| -OutputDir Data/Models |

Цялата команда ще изглежда така:

|  |
| --- |
| Scaffold-DbContext -Connection "Server=(localdb)\MSSQLLocalDB;Database=SoftUni;Integrated Security=True;" -Provider Microsoft.EntityFrameworkCore.SqlServer -OutputDir Data/Models |



Изпълнете **цялата команда** на **един ред**.

**Променяне на генерирания EF модел на данни**

Entity Framework Core успешно **мапна схемата на базата данни към C# класове**. Но не е достатъчно добър с имената – всички класове са в **множествено число**.

* Използвайте **Solution Explorer** във Visual Studio, за да преместите класа **SoftUniContext** от **Models** в папката **Data** и да преименувате правилно всичките ни класове.
* Използвайте **right click →** [**Rename**] или [**F2**] и натиснете [**OK**] в този **изскачащ прозорец** след всеки клас:



По този начин Visual Studio ще **преименува** **класовете навсякъде**, където са използвани.

Резултатът трябва да изглежда така:



Не забравяйте да коригирате namespace-a на **SoftUniContext**, след като го преместите, и добавете референция към namespace-а на **Models**:

Уверете се, че вашите namespace-и са **същите** като тези:

|  |
| --- |
| SoftUni  SoftUni.Data  SoftUni.Models |

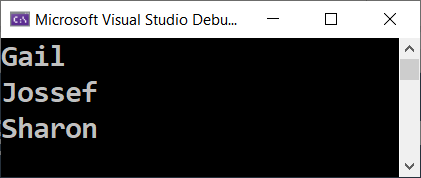
### Премахнете ненужните NuGet пакети

И накрая, можем да изчистим пакетите, които вече няма да използваме, от GUI на package manager-а или като изпълним тези команди:

|  |
| --- |
| Uninstall-Package Microsoft.EntityFrameworkCore.Tools -r  Uninstall-Package Microsoft.EntityFrameworkCore.SqlServer.Design -RemoveDependencies |

## Намерете служители с длъжност

Създайте **public static string FindEmployeesWithJobTitle(SoftUniContext context)** в класа StartUp, за да отпечатате **First Name** на всички служители с **Job Title**, равна на **“Design Engineer”**.



### Подсказка

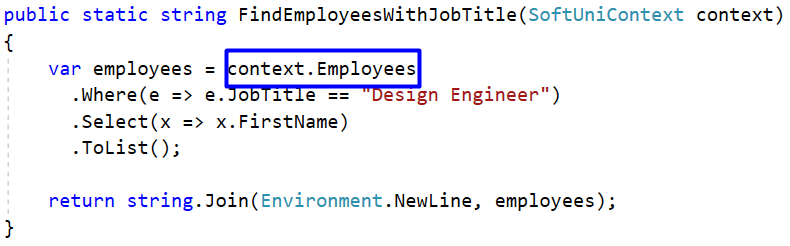
Your **class structure** should follow the provided **project skeleton**, like this:

**Структурата** на вашия **клас** трябва да следва предоставения скелет на проекта, като този:

Graphical user interface, text, application, email

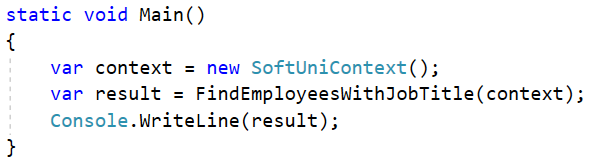
Description automatically generated

Вземете всички служители и ги **филтрирайте** с помощта на **context.Employees**. След това изберете само **First Name** на всеки служител и използвайте **String.Join()**, за да върнете масива от имена като стринг към метода.

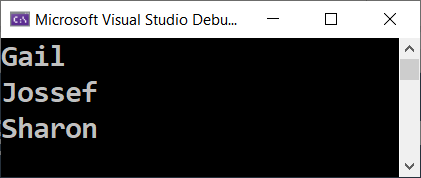


### Изпълнете кода си на конзолата

Извикайте метода **FindEmployeesWithJobTitle(SoftUniContext context)** от входната точка на приложението Main():



Натиснете **[Ctrl+F5]**, за да стартирате приложението. Проверете дали резултатът на конзолата е правилен:



### Качете своя код в Judge

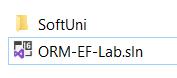
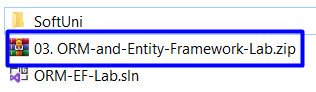
**Save your files** in Visual Studio. Open your project’s source folder:

**Запазете вашите файлове** във Visual Studio. Отворете source папката на вашия проект:

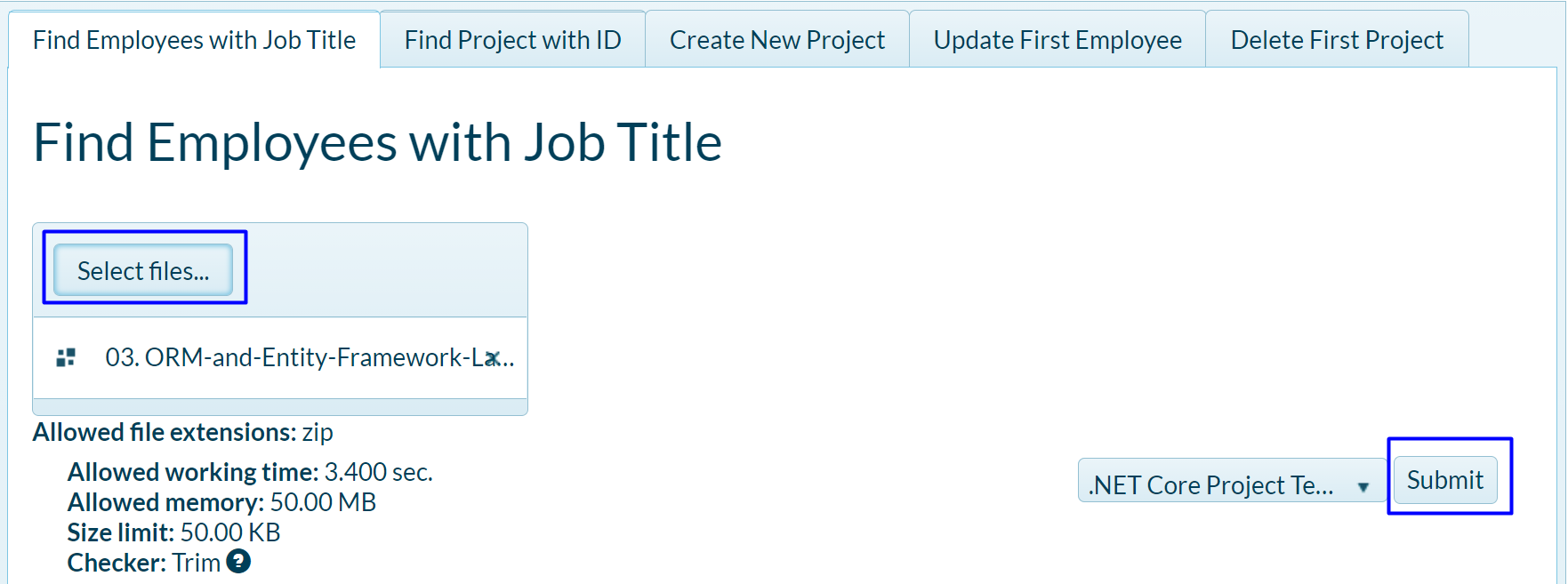
Graphical user interface, application

Description automatically generated

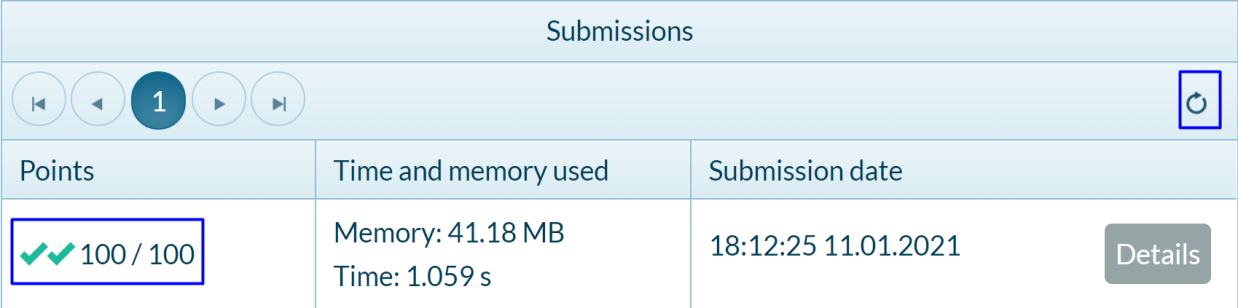
Delete the **“**bin**”** and **“**obj**”** folders from the **SoftUni** folder and create a **ZIP archive** of your solution:

**Submit** the ZIP archive file in SoftUni judge:

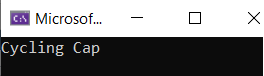


You should get 100 / 100 score:

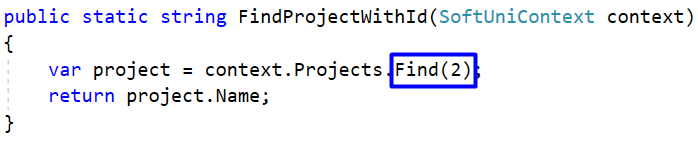


## Find Project with ID

Again, use the **context** and get all **Projects** from it. Use **.Find()** method to find the project with **ID 2** and return the **Name** of the project.



### Hints

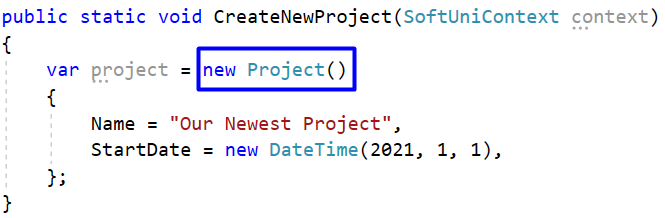


## Create New Project

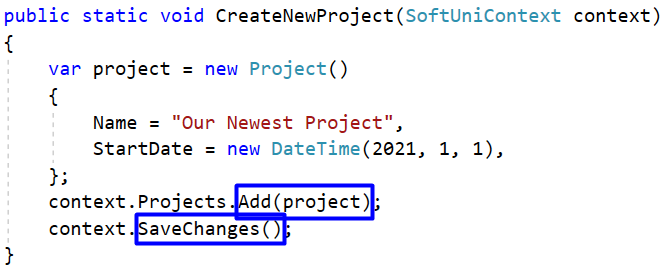
Your task is to create a **new Project** in the **Projects** table.

### Hints

To create a new database **row** use the **.Add()** method of the corresponding **DbSet**. First, create a new **Project** **object** and give values to **Name** and **StartDate** properties.



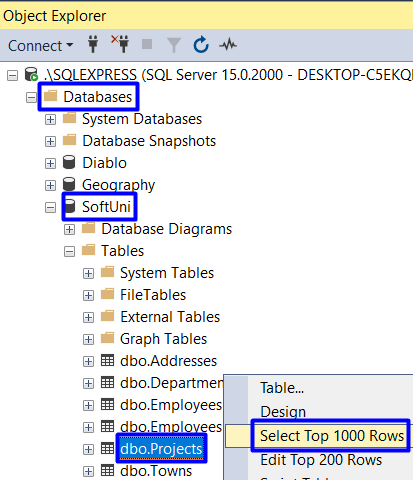
Then, add the object to the **DbSet** and do not forget to **save changes** the following way:



**Run** the app. There is no result displayed on the console.

### Check the Result in the DB

In order to check the result, go to **SQL Server Management Studio -> Object Explorer -> Databases -> SoftUni -> dbo.Projects**. **Right-click** on it and choose **Select Top 1000 Rows**.



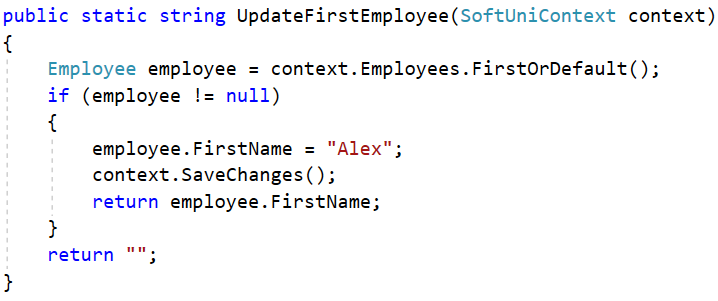
Scroll down to the **last entity**. It should be the one we added using a C# command in Visual Studio.



## Update First Employee

Get the **first employee** using **.FirstOrDefault()** method and change their **First Name** to **“Alex”**. Do not forget to **save changes**! In case there are no employees, return **empty string** to the method, else return the changed employee’s first name.

### Hints



### Check result in the DB

This is the database entity **before** the code execution:



After the code execution, the entity **should be changed**:

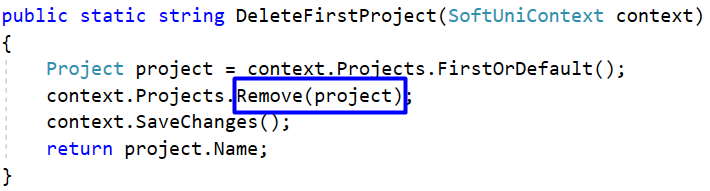


## Delete First Project

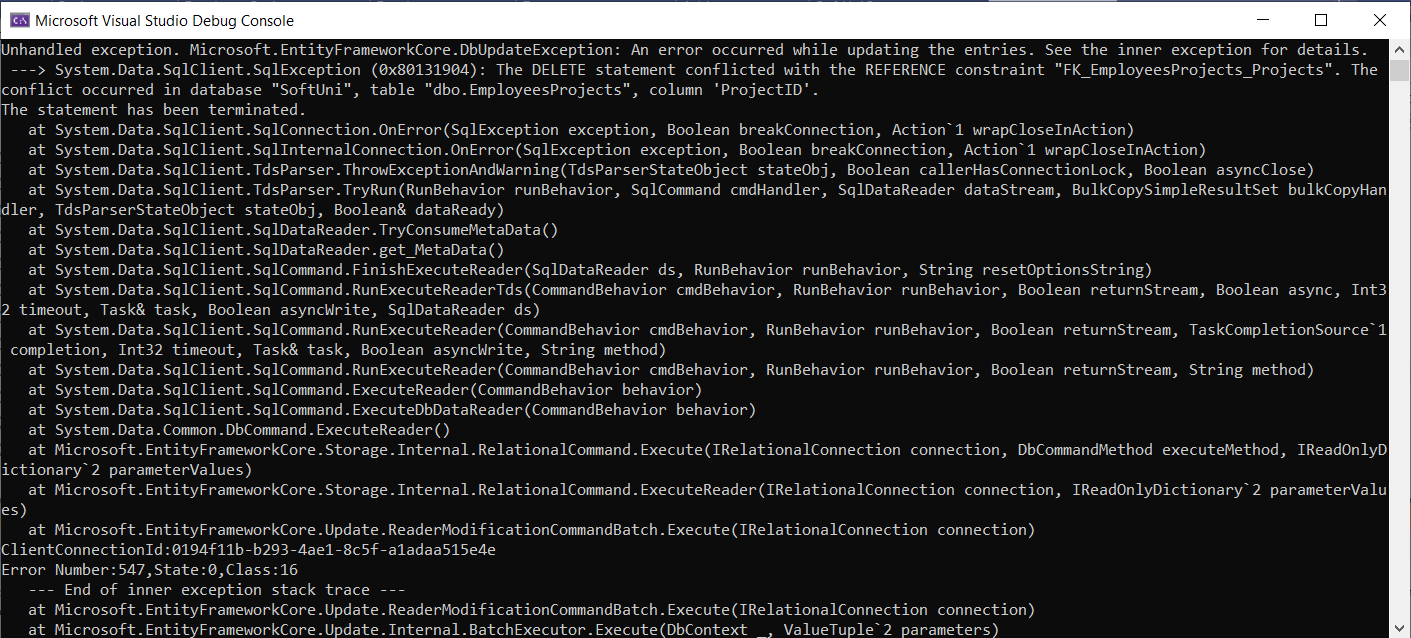
Get the **first project** and delete it using the **.Remove()** method. Do not forget to **save changes**! The entity we should remove is the following:



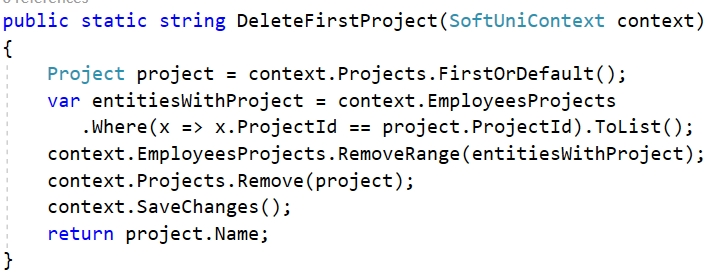
### Hints



However, when the program is executed an **error message** appears.



The reason for the error is that the **EmployeesProjects table** in the SoftUni DB contains a **ProjectID column**. So, entities from the Projects table **cannot be deleted** that way because some entities in the EmployeesProjects table contain the id of the project entity we want to delete. To solve that issue we may **first delete all enities** from the **EmployeesProjects** table, which contain our **ProjectId** (in our case with ProjectId=**1**). The command is the following:

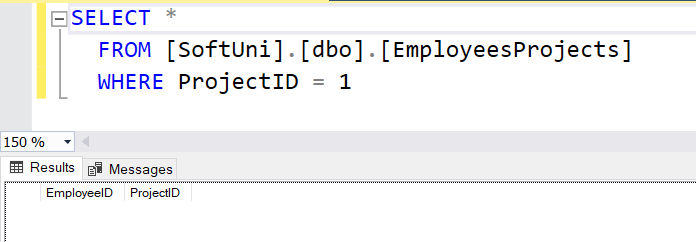


### Check result in the DB

Execute the program and see the result in the **Projects** table in the SoftUni DB.

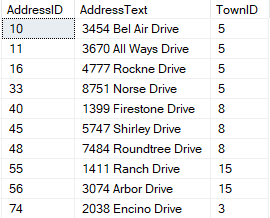
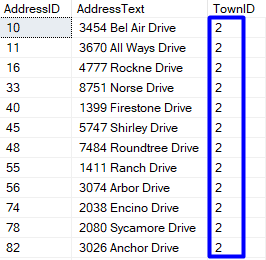


You can also check the **EmployeesProjects** table. Now it does not contain entities with **ProjectId = 1**.

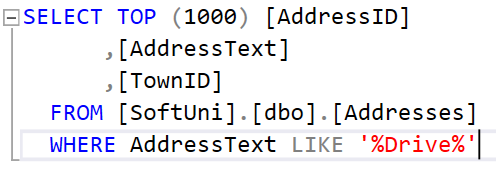


## Update Addresses

Write a **method** to update **TownId** to **2** for all **Addresses** with **AddressText**, containing the word **“Drive”**.

You can check the result in the **SoftUni DB** with this command:



The method **UpdateAddresses(SoftUniContext context)** should return the **count** of **changed** addresses, converted to **string**.

