

Connectivity From CodefirstApproach to Sql Server with net core

Step 1

Install Two Packages from NuGet Packager

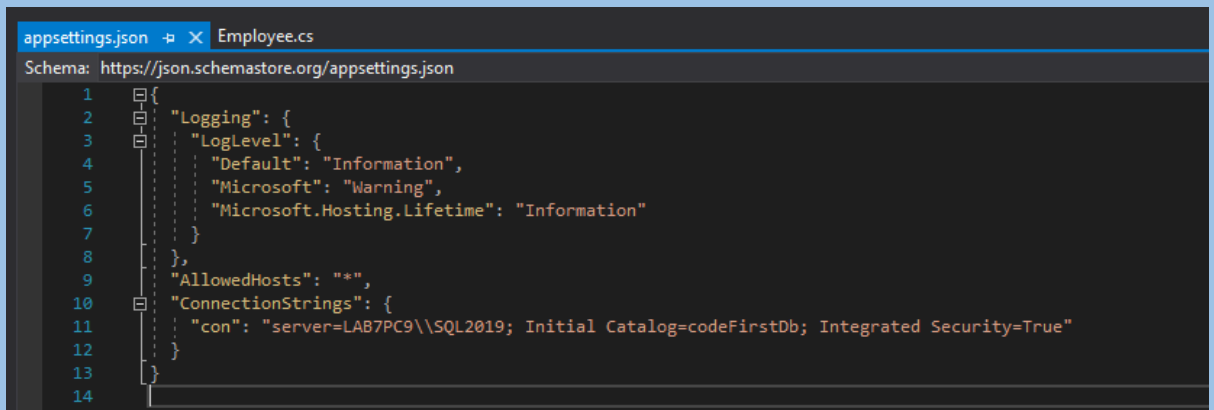
- Sql Server
- Tools

Step 2

- Open Sql Server Management Studio
- Create database
- You don't have to create table

Step 3

Go to Appsetting.json File



```
appsettings.json Employee.cs
Schema: https://json.schemastore.org/appsettings.json
1  {
2    "Logging": {
3      "LogLevel": {
4        "Default": "Information",
5        "Microsoft": "Warning",
6        "Microsoft.Hosting.Lifetime": "Information"
7      }
8    },
9    "AllowedHosts": "*",
10   "ConnectionStrings": {
11     "con": "server=LAB7PC9\\SQL2019; Initial Catalog=codeFirstDb; Integrated Security=True"
12   }
13 }
14
```

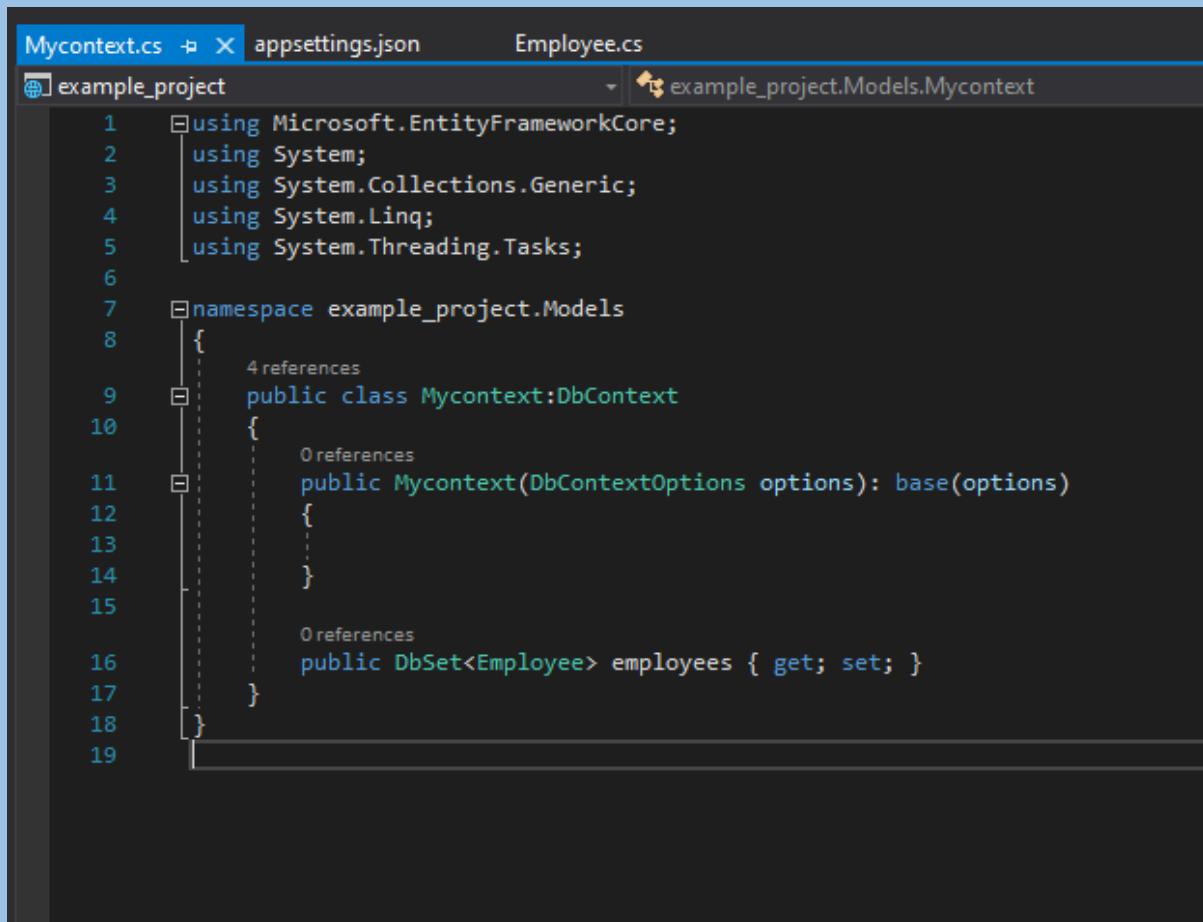
Step 4

Create model class name Employee

```
appsettings.json Employee.cs [X]
example_project example_project.Models.Employee
1 using System;
2 using System.Collections.Generic;
3 using System.ComponentModel.DataAnnotations;
4 using System.Linq;
5 using System.Threading.Tasks;
6
7 namespace example_project.Models
8 {
9     1 reference
10     public class Employee
11     {
12         [Key]
13         0 references
14         public int id { get; set; }
15
16         [Required]
17         0 references
18         public string EmployeeName { get; set; }
19
20         [Required]
21         0 references
22         public string EmployeeEmail { get; set; }
23
24         [Required]
25         0 references
26         public string EmployeePassword { get; set; }
27     }
28 }
```

Step 5

Create another model class name Mycontext

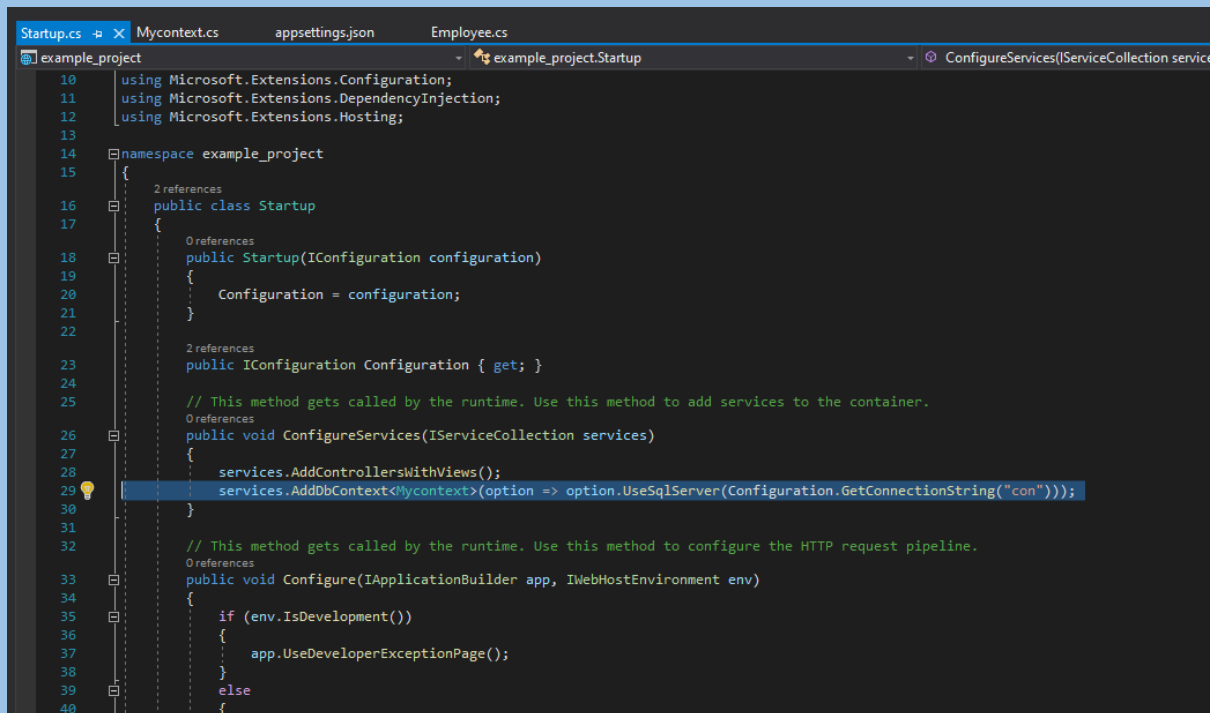


The screenshot shows the Visual Studio IDE with the 'Mycontext.cs' file open. The file is located in the 'example_project.Models' namespace. The code defines a 'Mycontext' class that inherits from 'DbContext'. It includes several using statements at the top and a DbSet property for 'employees'.

```
1 using Microsoft.EntityFrameworkCore;
2 using System;
3 using System.Collections.Generic;
4 using System.Linq;
5 using System.Threading.Tasks;
6
7 namespace example_project.Models
8 {
9     4 references
10     public class Mycontext:DbContext
11     {
12         0 references
13         public Mycontext(DbContextOptions options): base(options)
14         {
15         }
16
17         0 references
18         public DbSet<Employee> employees { get; set; }
19     }
20 }
```

Step 6

Go to Startup.cs File



The screenshot shows the Visual Studio IDE with the 'Startup.cs' file open. The file is part of a project named 'example_project'. The code defines a 'Startup' class within the 'example_project' namespace. It includes using statements for 'Microsoft.Extensions.Configuration', 'Microsoft.Extensions.DependencyInjection', and 'Microsoft.Extensions.Hosting'. The 'Startup' class has a constructor that takes an 'IConfiguration' object and assigns it to a 'Configuration' property. It also has a 'ConfigureServices' method that adds controllers and a database context. The 'Configure' method is also present, which configures the HTTP request pipeline based on the environment (Development or otherwise).

```
10 using Microsoft.Extensions.Configuration;
11 using Microsoft.Extensions.DependencyInjection;
12 using Microsoft.Extensions.Hosting;
13
14 namespace example_project
15 {
16     2 references
17     public class Startup
18     {
19         0 references
20         public Startup(IConfiguration configuration)
21         {
22             Configuration = configuration;
23         }
24
25         2 references
26         public IConfiguration Configuration { get; }
27
28         // This method gets called by the runtime. Use this method to add services to the container.
29         0 references
30         public void ConfigureServices(IServiceCollection services)
31         {
32             services.AddControllersWithViews();
33             services.AddDbContext<Mycontext>(option => option.UseSqlServer(Configuration.GetConnectionString("con")));
34         }
35
36         // This method gets called by the runtime. Use this method to configure the HTTP request pipeline.
37         0 references
38         public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
39         {
40             if (env.IsDevelopment())
41             {
42                 app.UseDeveloperExceptionPage();
43             }
44             else
45             {
46                 // ...
47             }
48         }
49     }
50 }
```

Step 7

Install Two Commands

Tools -> nuget package manager -> package Manager Console

- 1) add-migration migrationName (will create migration folder in your project)
- 2) Update-database (will update database in SSMS)
- 3)

Open your SSMS and Check database