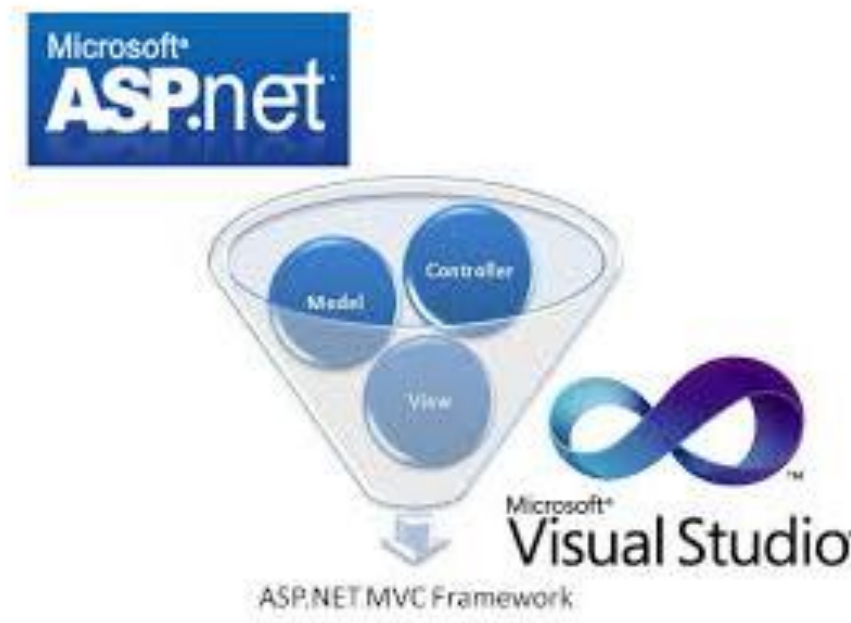
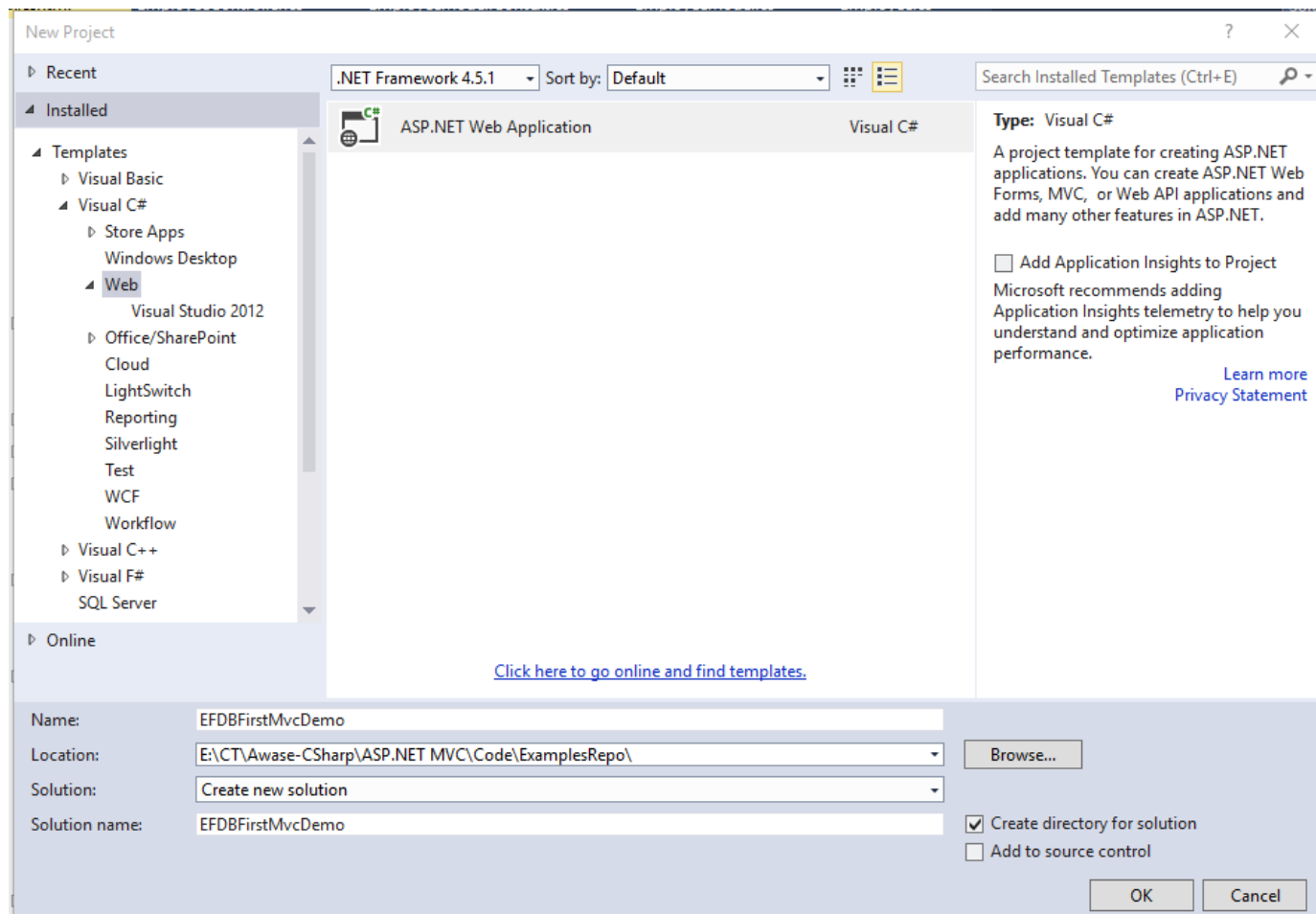


C#

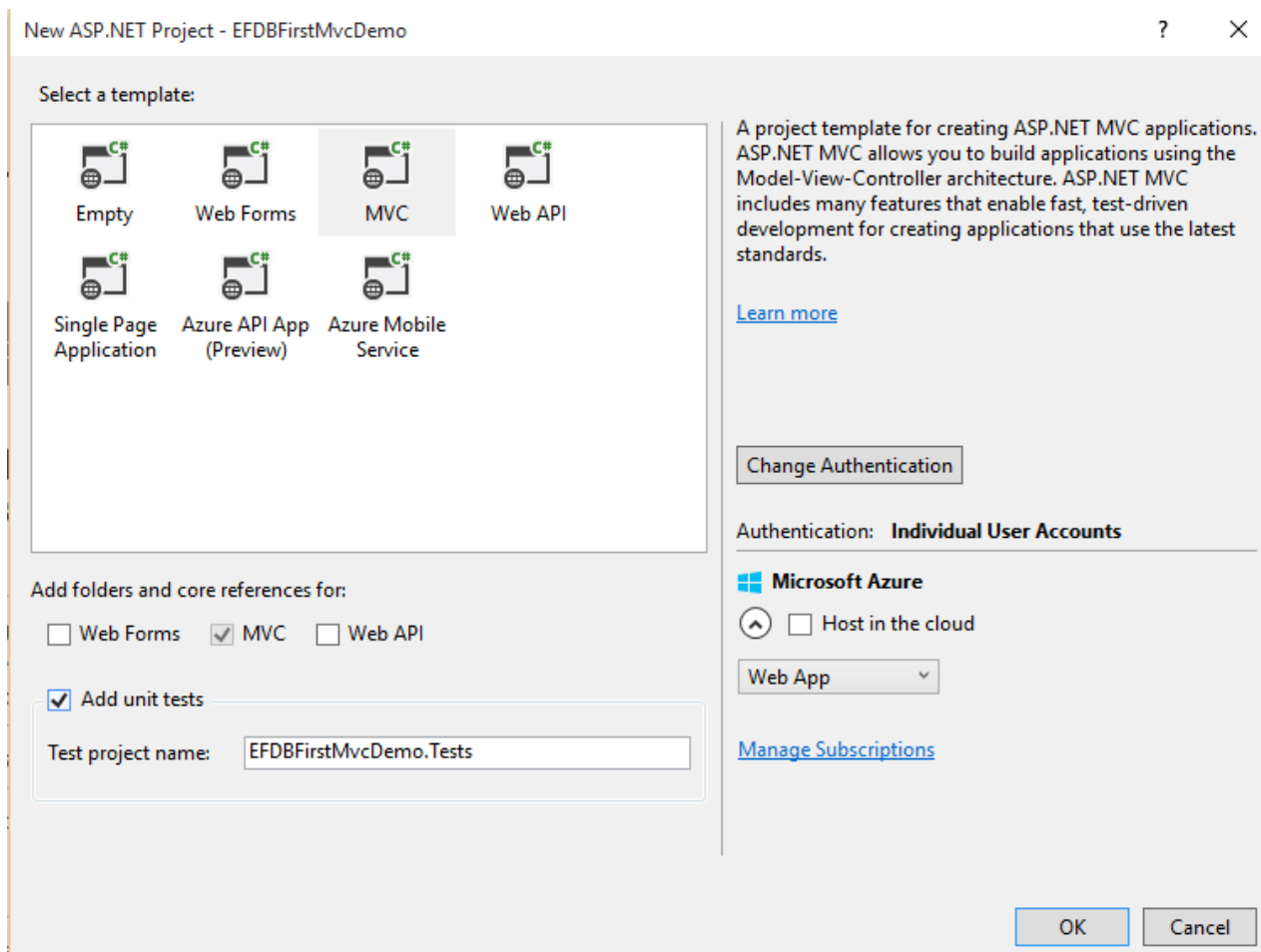


4. WORKING WITH DATA IN ASP.NET MVC USING ENTITY FRAME WORK DB FIRST

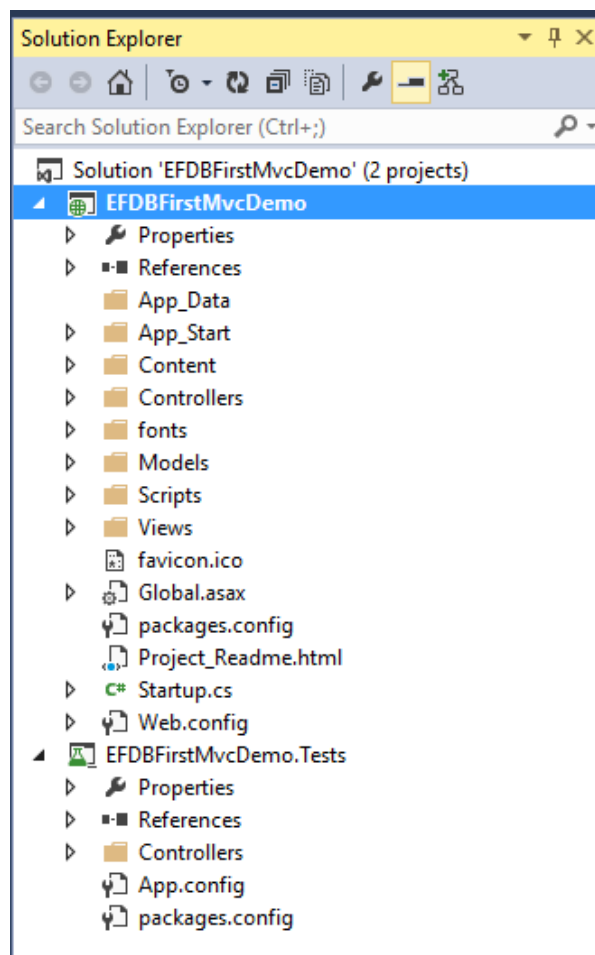
1. Create ASP.NET Web Application



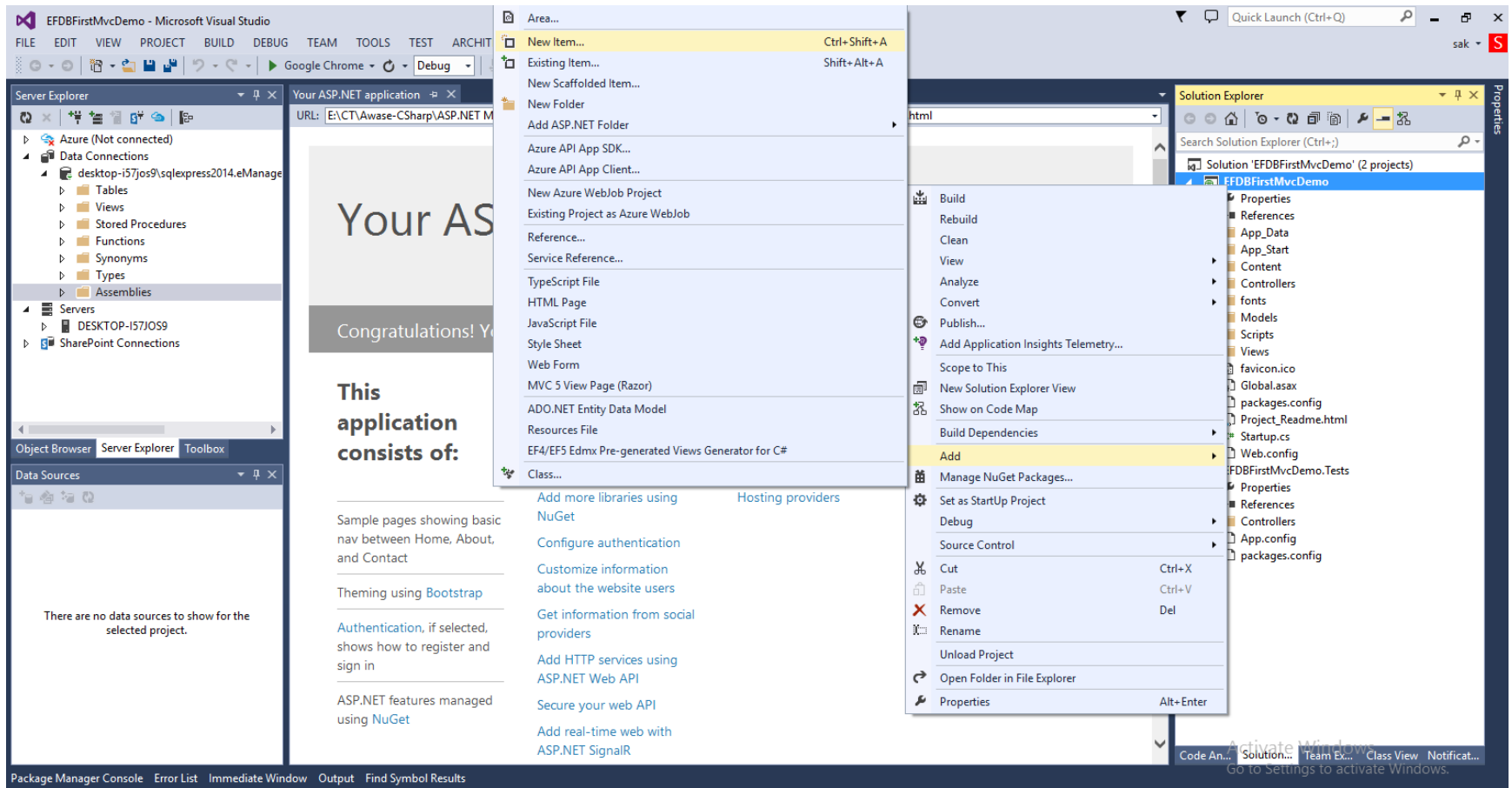
2. Select MVC Web Application with Unit Testing



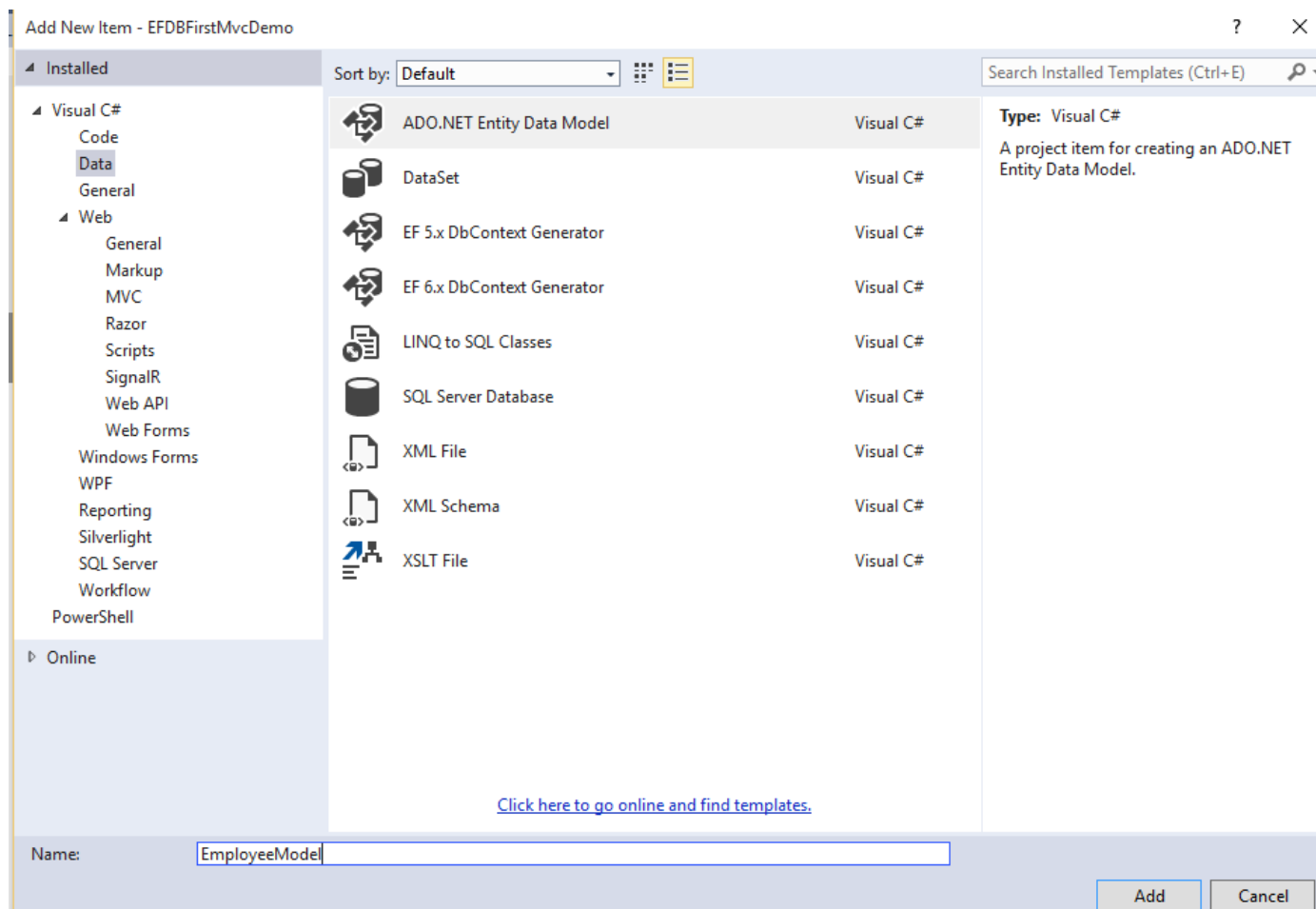
3. Scaffolding MVC Application



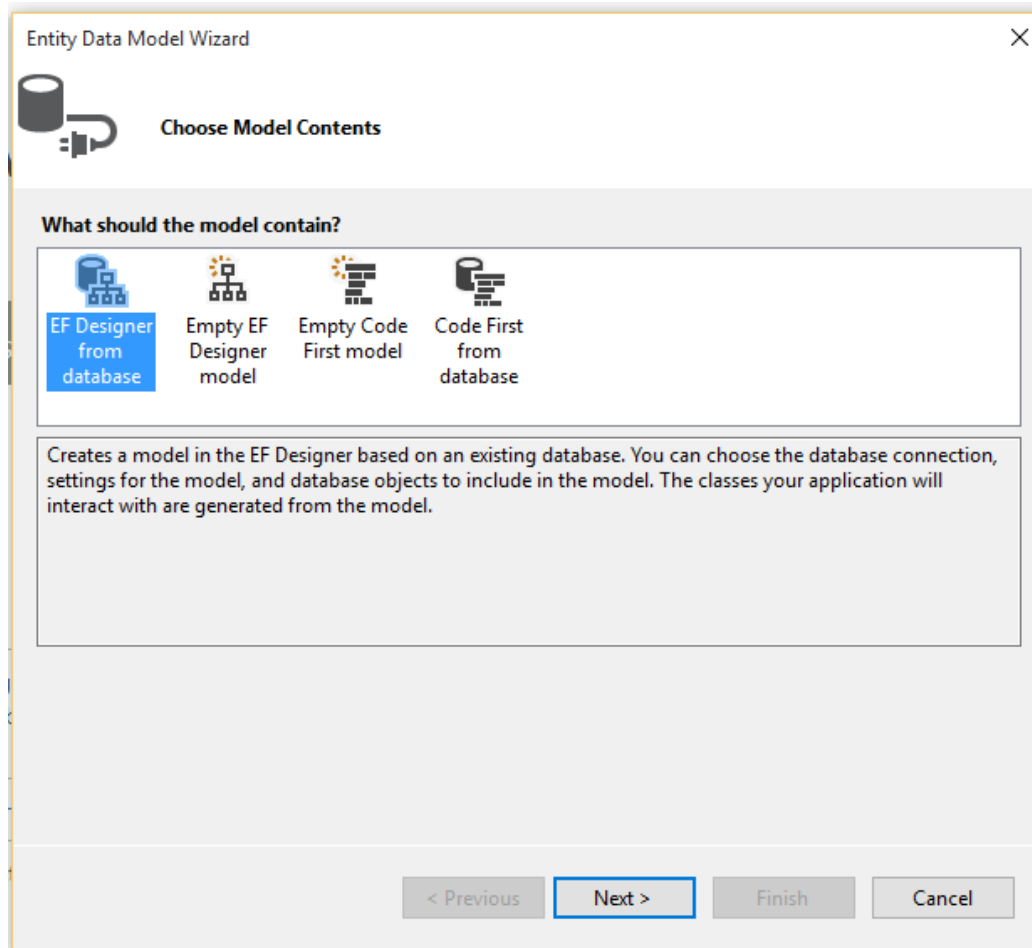
4. Add -> New Item



5. Create ADO.NET ENTITY DATA MODEL




6.EF Designer from Database



7.DATA CONNECTION

Entity Data Model Wizard ×

 Choose Your Data Connection

Which data connection should your application use to connect to the database?

desktop-i57jos9\sqlexpress2014.eManager.Web.Infrastructure.DepartmentDb ▾ New Connection...

This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

☐ No, exclude sensitive data from the connection string. I will set it in my application code.

☐ Yes, include the sensitive data in the connection string.

Connection string:

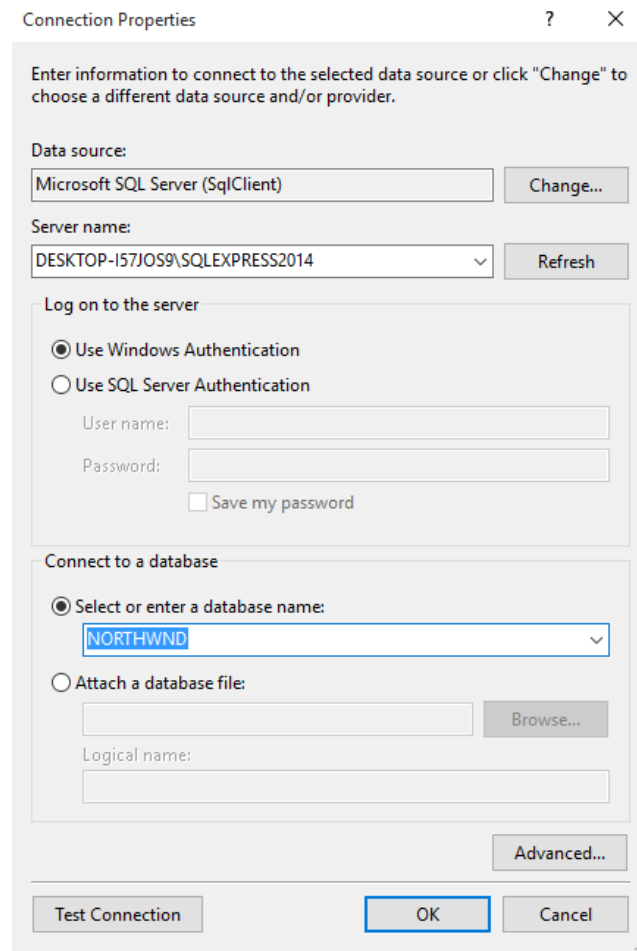
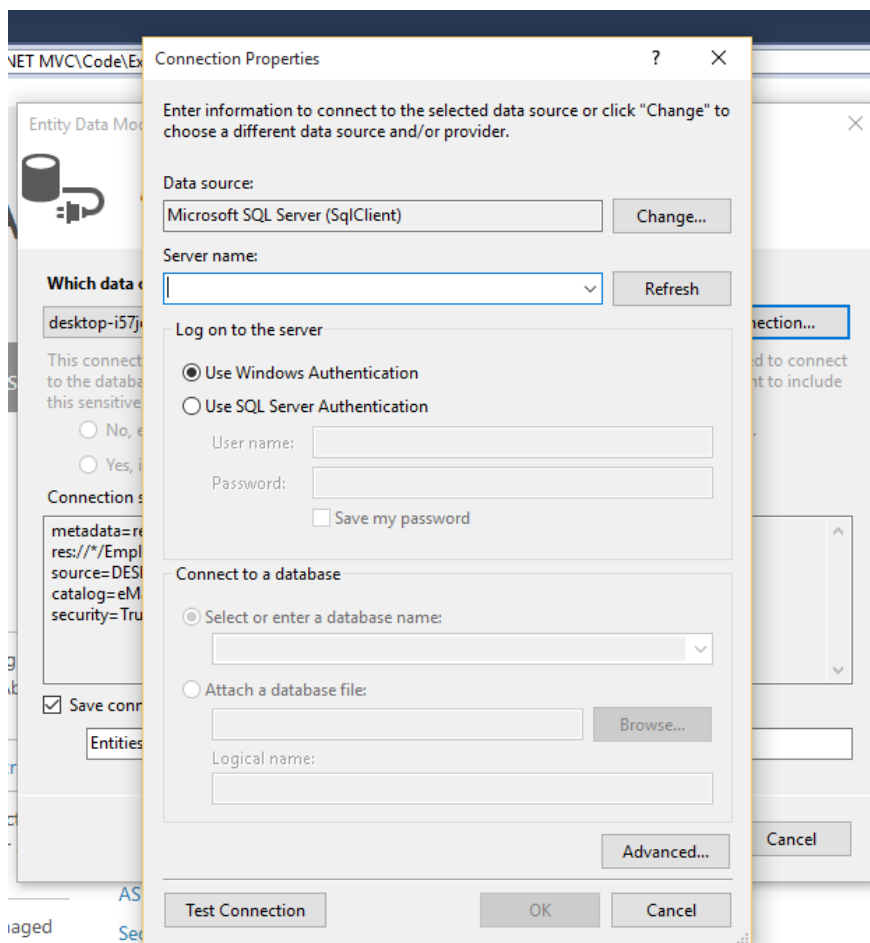
```
metadata=res://*/EmployeeModel.csdl|res://*/EmployeeModel.ssdl|
res://*/EmployeeModel.msl;provider=System.Data.SqlClient;provider connection string="data
source=DESKTOP-I57JOS9\SQLEXPRESS2014;initial
catalog=eManager.Web.Infrastructure.DepartmentDb;integrated
security=True;pooling=False;MultipleActiveResultSets=True;App=EntityFramework"
```

☒ Save connection settings in Web.Config as:

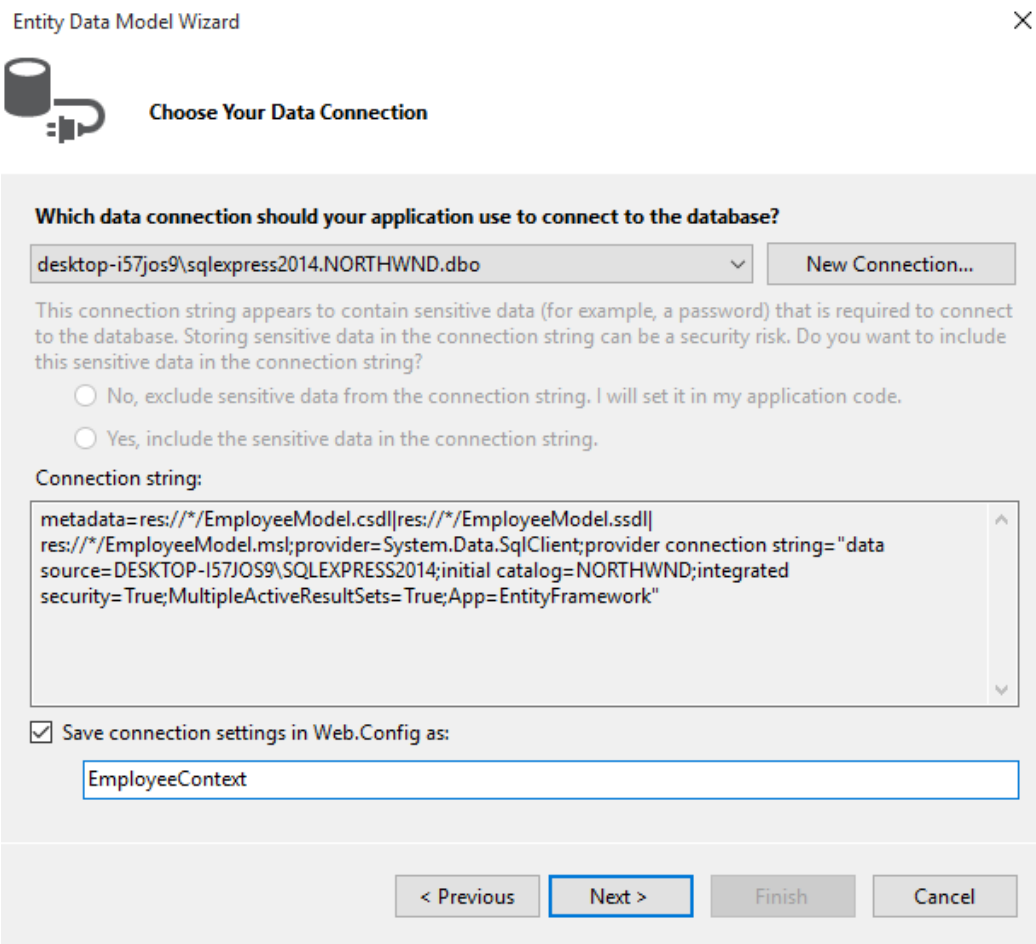
Entities

< Previous Next > Finish Cancel

8. DEFINE CONNECTION PROPERTIES



9. Post Connection Setting



The image shows a screenshot of the 'Entity Data Model Wizard' window, specifically the 'Choose Your Data Connection' step. The window has a title bar with 'Entity Data Model Wizard' and a close button. Below the title bar is a database icon and the text 'Choose Your Data Connection'. The main content area asks 'Which data connection should your application use to connect to the database?'. There is a dropdown menu showing 'desktop-i57jos9\sqlexpress2014.NORTHWND.dbo' and a 'New Connection...' button. Below this, a warning message states: 'This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?'. There are two radio buttons: 'No, exclude sensitive data from the connection string. I will set it in my application code.' and 'Yes, include the sensitive data in the connection string.'. Below the radio buttons is a section labeled 'Connection string:' with a text area containing the following text: 'metadata=res://*/EmployeeModel.csdl|res://*/EmployeeModel.ssdl|res://*/EmployeeModel.msl;provider=System.Data.SqlClient;provider connection string="data source=DESKTOP-I57JOS9\SQLEXPRESS2014;initial catalog=NORTHWND;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework"'. At the bottom, there is a checkbox labeled 'Save connection settings in Web.Config as:' which is checked. Below the checkbox is a text box containing 'EmployeeContext'. At the very bottom, there are four buttons: '< Previous', 'Next >', 'Finish', and 'Cancel'.

Entity Data Model Wizard

Choose Your Data Connection

Which data connection should your application use to connect to the database?

desktop-i57jos9\sqlexpress2014.NORTHWND.dbo New Connection...

This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

☐ No, exclude sensitive data from the connection string. I will set it in my application code.

☐ Yes, include the sensitive data in the connection string.

Connection string:

```
metadata=res://*/EmployeeModel.csdl|res://*/EmployeeModel.ssdl|
res://*/EmployeeModel.msl;provider=System.Data.SqlClient;provider connection string="data
source=DESKTOP-I57JOS9\SQLEXPRESS2014;initial catalog=NORTHWND;integrated
security=True;MultipleActiveResultSets=True;App=EntityFramework"
```


☒ Save connection settings in Web.Config as:

EmployeeContext


< Previous Next > Finish Cancel


10. Choose Database Objects and Settings











Entity Data Model Wizard ×

 Choose Your Database Objects and Settings

Which database objects do you want to include in your model?

☒  Tables

☒  dbo

- ☐  Categories
- ☐  CustomerCustomerDemo
- ☐  CustomerDemographics
- ☐  Customers
- ☒  Employees
- ☐  EmployeeTerritories
- ☐  Order Details
- ☐  Orders
- ☐  Products
- ☐  Region

☒ Pluralize or singularize generated object names

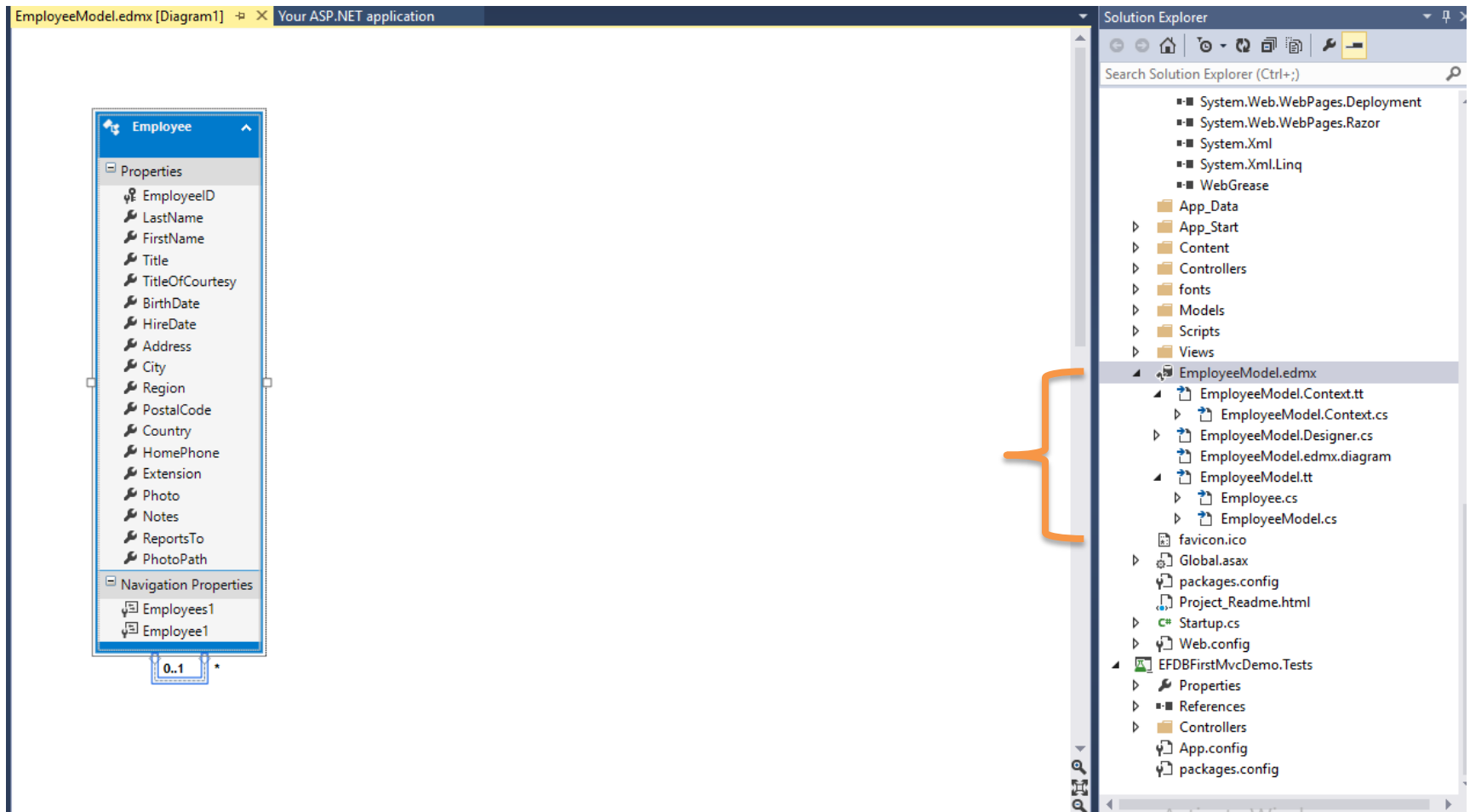
☒ Include foreign key columns in the model

☒ Import selected stored procedures and functions into the entity model

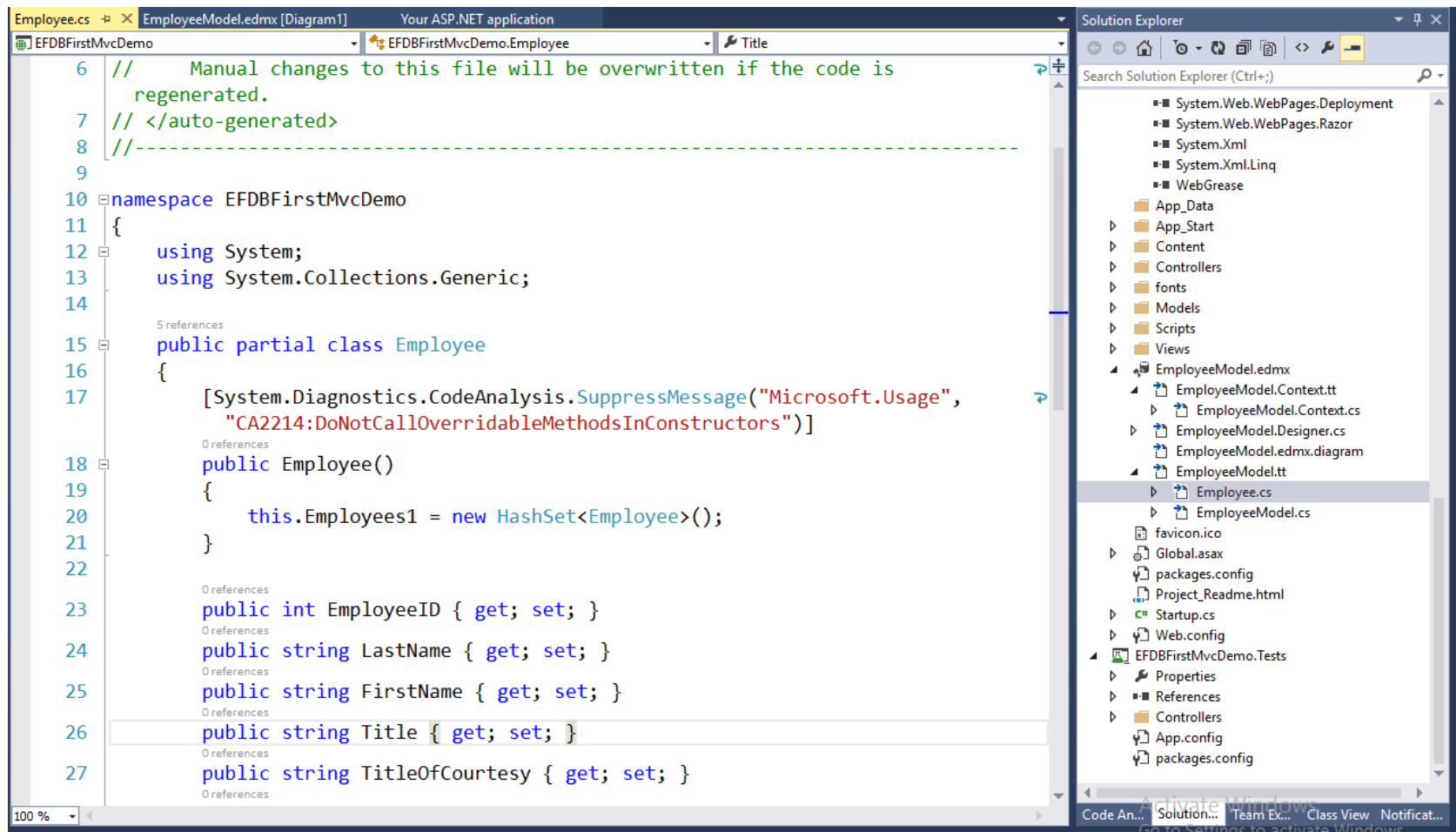
Model Namespace:

< Previous Next > Finish Cancel

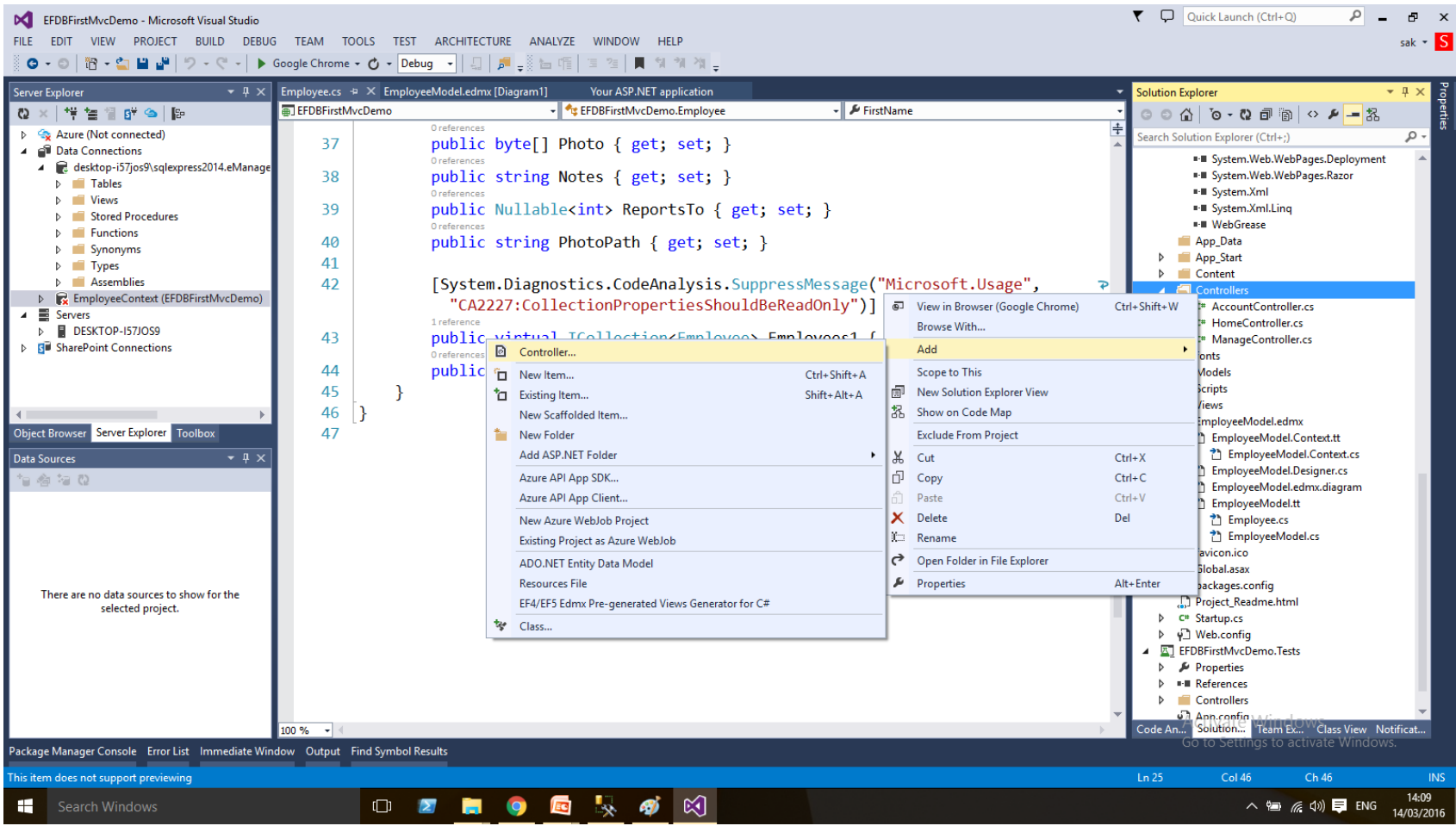
11. Successful Creation of Entity Model



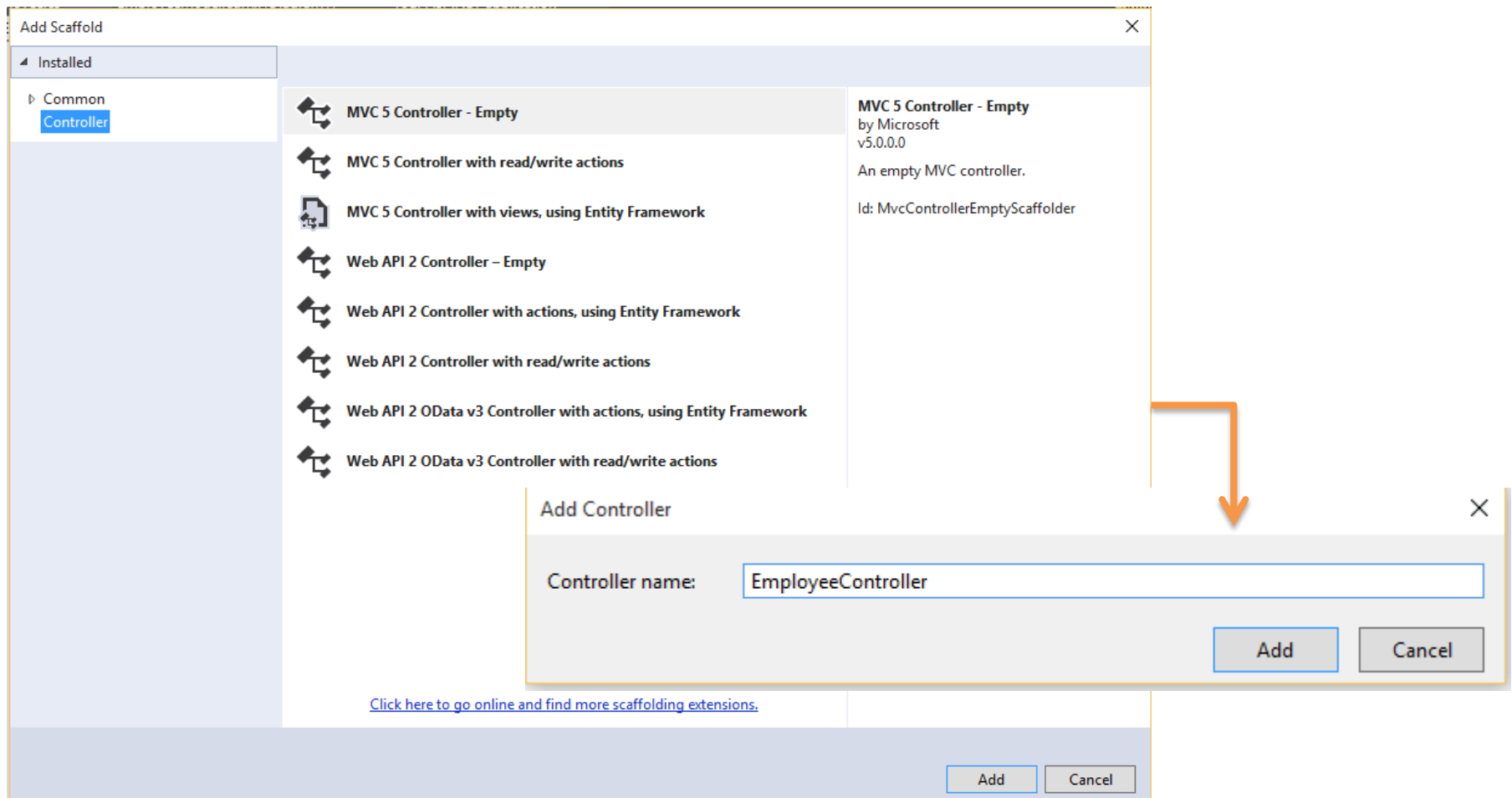
11. Successful Creation of Entity Model



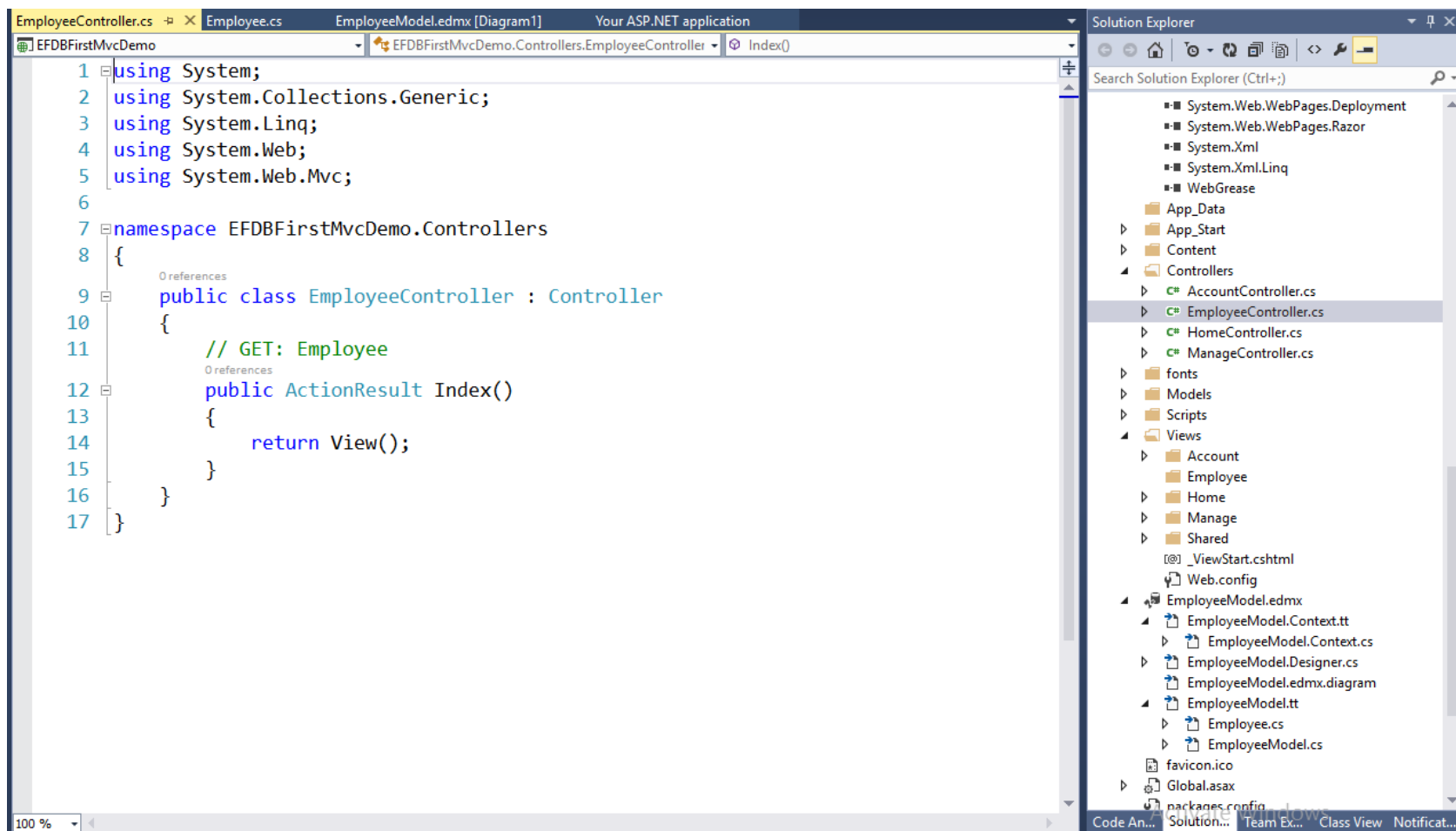
12. Create EmployeeController



13. Empty Controller



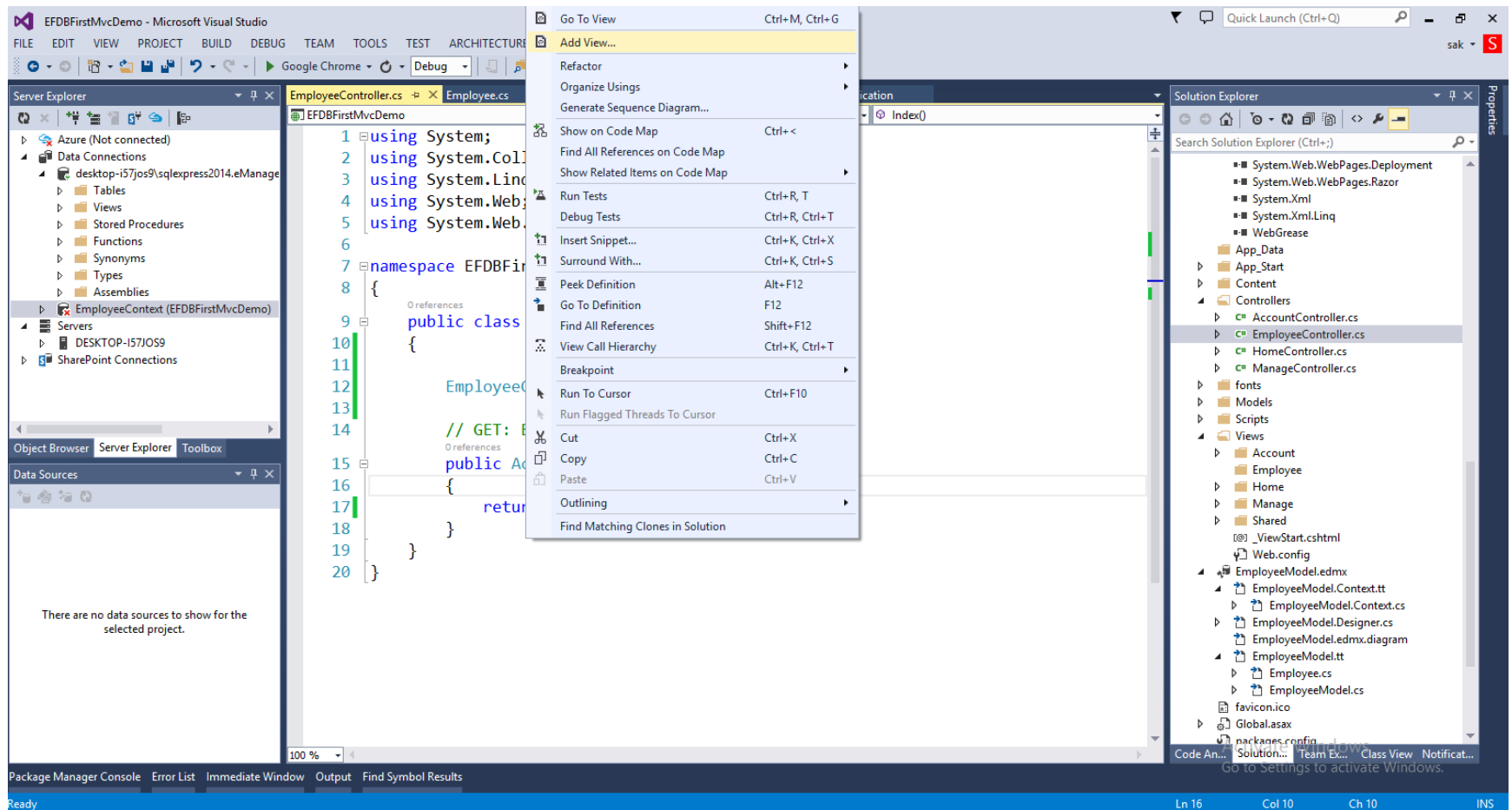
14.EmployeeController



15. Retrieve the List of Employee in Controller

```
-----  
public class EmployeeController : Controller  
{  
  
    EmployeeContext db = new EmployeeContext();  
  
    // GET: Employee  
    0 references  
    public ActionResult Index()  
    {  
        return View(db.Employees.ToList());  
    }  
}
```

16. Generate View for Index



17. Index View

Add View [X]

View name:

Template:

Model class:

Data context class:

Options:

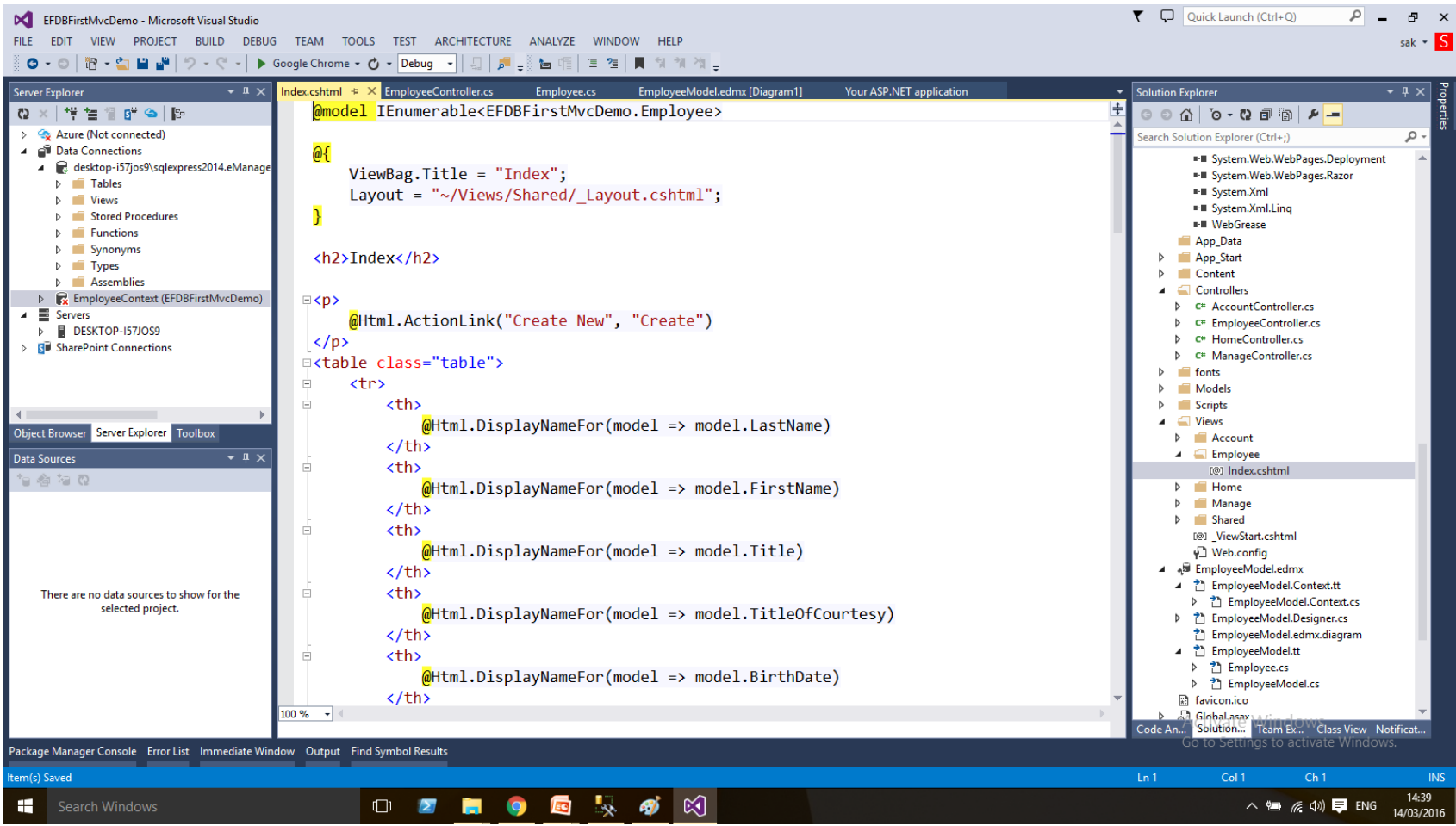
☐ Create as a partial view

☒ Reference script libraries

☒ Use a layout page:

(Leave empty if it is set in a Razor _viewstart file)

18. Index View Scaffolding Template



19. Result

Application name Home About Contact Register Log in

List of All Employees

[Create New](#)

LastName	FirstName	Title	TitleOfCourtesy	BirthDate	HireDate	Address	City	Region	PostalCode	Country	HomePhone	Extension	Photo
Davolio	Nancy	Sales Representative	Ms.	08/12/1948 00:00:00	01/05/1992 00:00:00	507 - 20th Ave. E. Apt. 2A	Seattle	WA	98122	USA	(206) 555-9857	5467	21284702000130140200330
Fuller	Andrew	Vice President, Dr. Sales		19/02/1952 00:00:00	14/08/1992 00:00:00	908 W. Capital Way	Tacoma	WA	98401	USA	(206) 555-9482	3457	21284702000130140200330

Activate Windows
Go to Settings to activate Windows.

ModelState

- ModelState is a property of a Controller, and can be accessed from those classes that inherit from [System.Web.Mvc.Controller](#).
- The ModelState represents a collection of name and value pairs that were submitted to the server during a POST.
- It also contains a collection of error messages for each value submitted.
- Despite its name, it doesn't actually know anything about any model classes, it only has names, values, and errors.
- Serves 2 purposes
 - To store the value submitted to the server
 - To store the validation errors associated with those values
- ModelState represents the submitted values and errors in said values during a POST

DataAnnotations

- Data Validation is a key aspect for developing web application. In ASP.NET MVC, we can apply validation to web application by using **Data Annotation Attribute classes to model class.**
- **Data Annotation attribute classes are present in System.ComponentModel.DataAnnotations namespace.**
- They help us to define rules to the model classes or properties for data validation and displaying suitable message to end users.

Annotation Type	Description
Data Type	Specify the datatype of a property
DisplayName	Specify the display name for a property
DisplayFormat	Specify the display format for a property like different format for Date property
Required	Specify a property as required
RegularExpression	Validate the value of a property by specified regular expression pattern
Range	Validate the value of a property within a specified range of values
StringLength	Specify min and max length for a string property
MaxLength	Specify max length for a string property
Bind	Specify fields to include or exclude when adding parameter or form values to model properties
ScaffoldColumn	Specify fields for hiding from editor forms

Data Annotations

Advantages

- Enable you to perform validation simply by adding one or more attributes – such as required or StringLength attribute – to a class property

MVC5 WITH EF CODE FIRST

Code First EF MVC5 Application

Step 1. Create MVC5 Application

Step 2. Create A folder Context

Step 3. Create a Model –

ModelName.cs

```
public class RealEstateProperty
{
    [Key]
    public int EstatePropertyId { get; set; }

    [Required]
    [DisplayName("Natural Address Code 30 digit Alphanumeric code")]
    public string NACCode { get; set; }

    public string EstatePropertyName { get; set; }

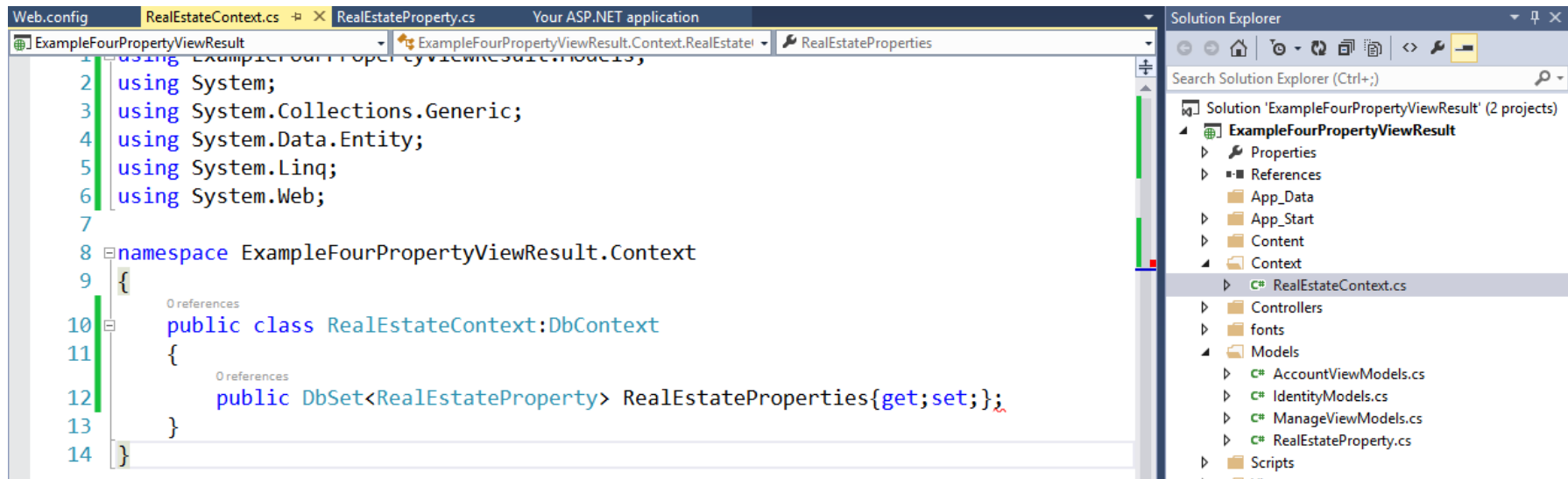
    public string OwnerName { get; set; }

    public string City { get; set; }

    public string Country { get; set; }

    public string Zip { get; set; }
```

- Step 4. Create a ModelContext class in Context Folder – ModelContext.cs



- Step 5. Add ConnectionString in web.config

```
<connectionStrings>
  <add name="DefaultConnection" connectionString="Data Source=(LocalDb)\v11.0;AttachDb
    providerName="System.Data.SqlClient" />
  <add name="RealEstateContext" connectionString="Data Source=(LocalDb)\v11.0;Attach
    providerName="System.Data.SqlClient" />
</connectionStrings>
```

- Step 6. Create a Controller

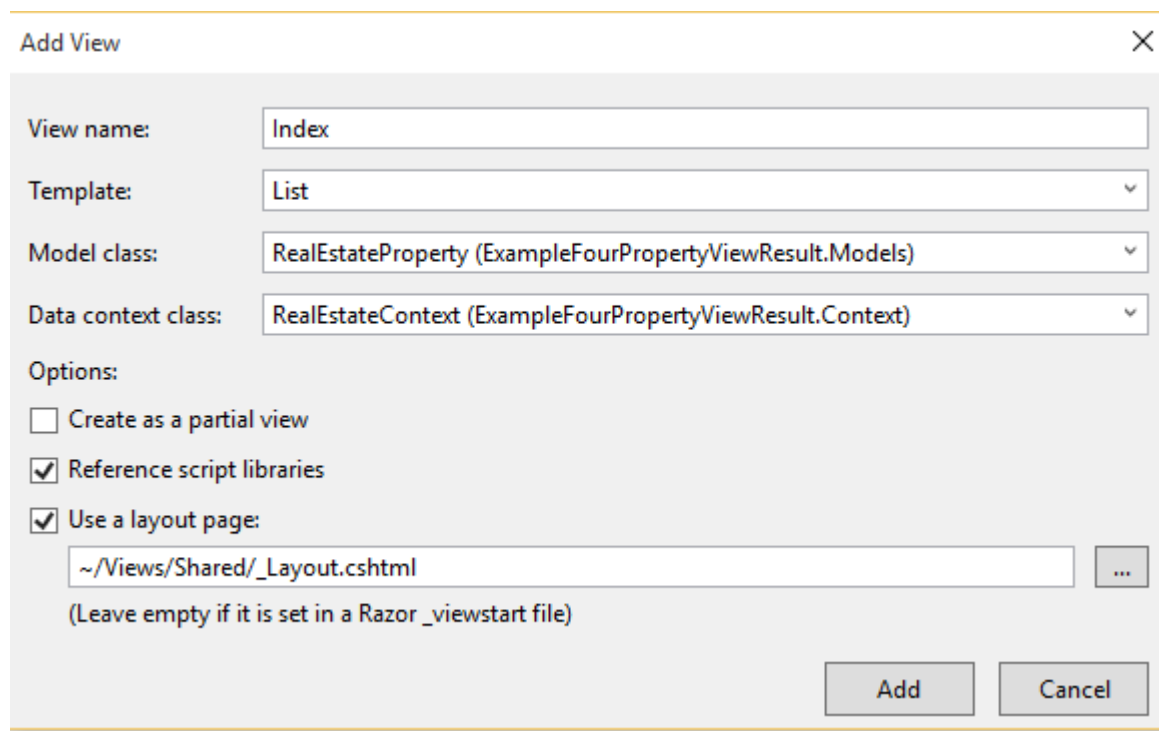
```
namespace ExampleFourPropertyViewResult.Controllers
{
    0 references
    public class RealEstatePropertyController : Controller
    {
        0 references
        // GET: RealEstateProperty
        public ActionResult Index()
        {
            return View();
        }
    }
}
```

- Step 7. Add DbContext to the Controller and return list to the conventional view

```
1 using ExampleFourPropertyViewResult.Context;
2 using System;
3 using System.Collections.Generic;
4 using System.Linq;
5 using System.Web;
6 using System.Web.Mvc;
7
8 namespace ExampleFourPropertyViewResult.Controllers
9 {
10     public class RealEstatePropertyController : Controller
11     {
12         RealEstateContext db = new RealEstateContext();
13         // GET: RealEstateProperty
14         public ActionResult Index()
15         {
16             return View(db.RealEstateProperties.ToList());
17         }
18     }
19 }
```

- Step 8. Build the application with ctrl+shift+B

➤ Step 9. Generate Index View



Add View [X]

View name:

Template:

Model class:

Data context class:

Options:

☐ Create as a partial view

☒ Reference script libraries

☒ Use a layout page:

...

(Leave empty if it is set in a Razor _viewstart file)

➤ Step 10. Run the Application – ctrl +f5

➤ Step 11. Upon first access to the Controller Action Method, We have a DB generated from **the RealEstateProperty Model**

Summary

- Demonstrated how to create EF DB First and Code First approaches
- Demonstrated the use of data annotations for validations