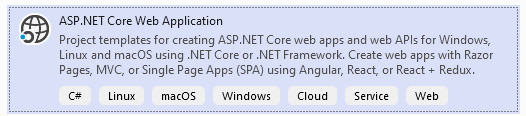
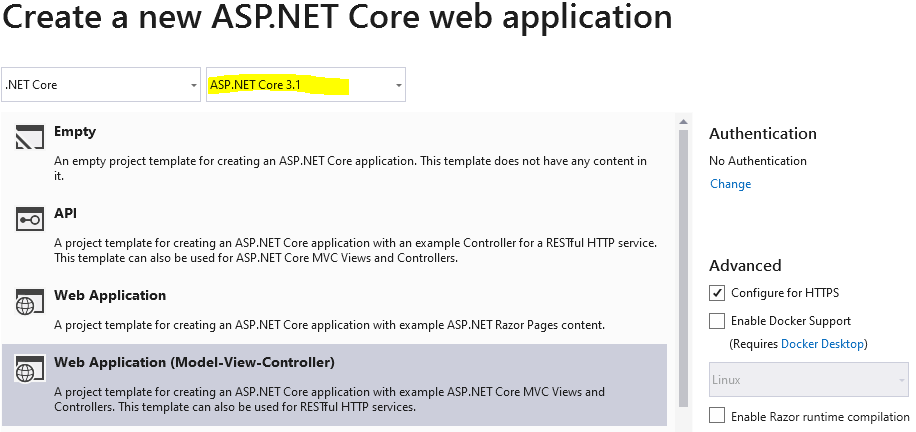
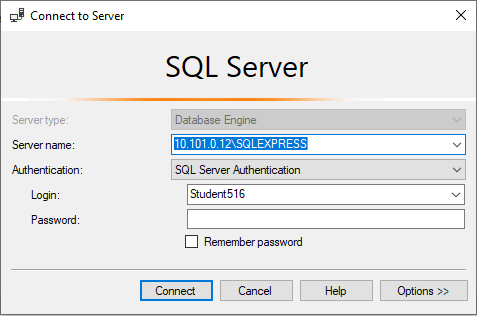
Setting up Entity Framework Core Tutorial

1. Install/Update Visual Studio 2019 to at least version 16.3 in order to support .NET Core 3.1 applications.
2. Create a project: ASP.NET Core MVC (Core 3.1) 

Next window:

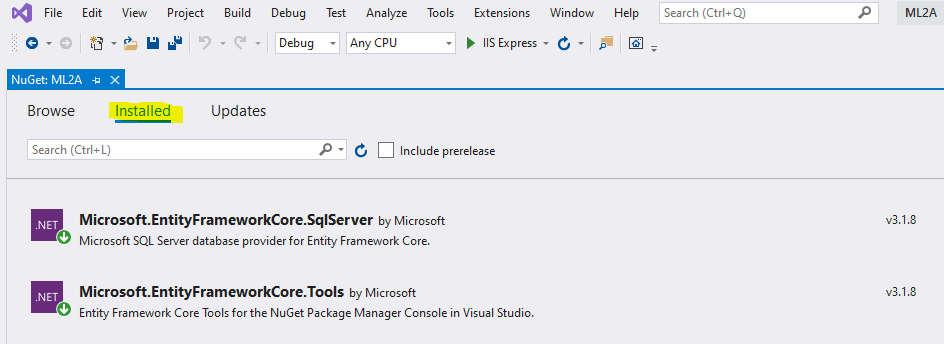


1. Because we are working off campus, we have to use a VPN to connect to JAC local servers. See Lea for VPN instructions.
2. Make sure you have a SQL Database and that you can connect to it.
   1. Download SQL Server Management Studio (SSMS)  
      <https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15>



* 1. Enter this information:   
     Username: Student516   
     Password: SecretSauce2020

1. Take the time to analyse the database AdventureWorksLT2017 and its tables.
2. Back in ASP.NET Application. Go ahead and Install Entity Framework Core from NuGet (version 3.1.x for all):
   1. See detailed instructions here: <https://www.entityframeworktutorial.net/efcore/install-entity-framework-core.aspx>
   2. Install:
      1. Microsoft.EntityFrameworkCore.SqlServer
      2. Microsoft.EntityFrameworkCore.Tools
3. Double check that the packages have been installed properly:



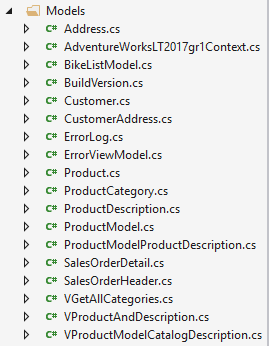
1. Scaffold all the AdventureWorksLT2017 tables as classes:
   1. Select View 🡪 Other Windows 🡪 Package Manager Console. Write:
      1. For remote DB: (copy and paste)

Scaffold-DbContext "Server=10.101.0.12\SQLEXPRESS;Database=AdventureWorksLT2017;User ID=Student516;Password=SecretSauce2020" Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models

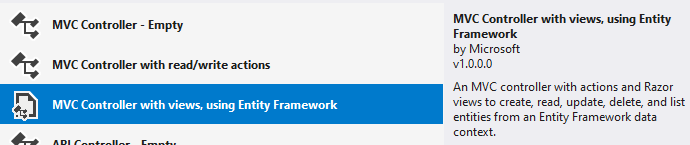
* + 1. For Local DB:

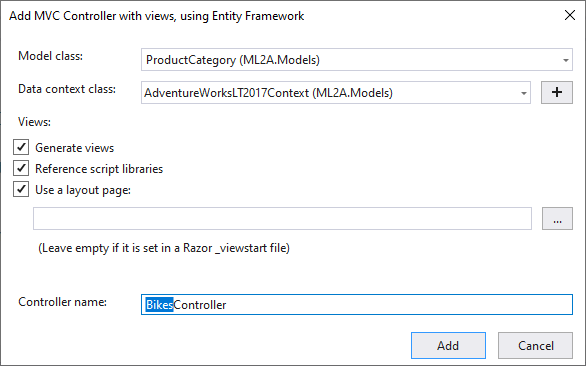
Scaffold-DbContext "Server=(localdb)\mssqllocaldb;Database=AdventureWorksLT2017;Trusted\_Connection=True;" Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models

1. You should see the classes created based on the tables and views from the database:

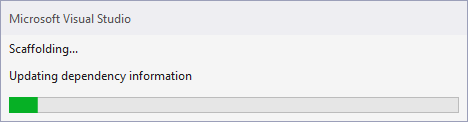


1. From Visual studio, run your application with IIS Express. If it does not work, run your project name (it uses Kestrel server).
2. Create a Controller with views using EntityFramework based on ProductCategory class.

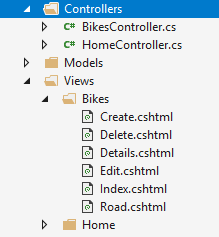




Scaffolding is the process of automatically creating the Controller with all the actions related to the CRUD, as well as the respective Views pages.

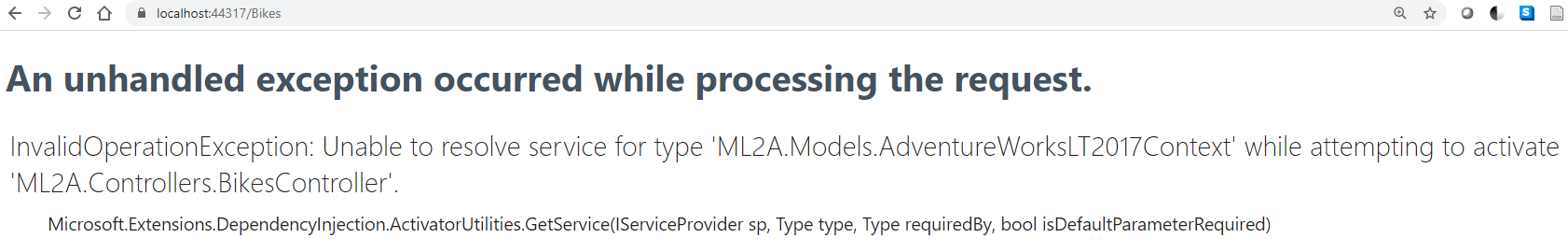


1. The scaffolding should produce the Controller and all the CRUD views.



1. Run your web application and go to the Bikes controller:

<https://localhost:44317/Bikes>

1. This error message will appear. 

This is because we need to add the ConnectionString pointing to the DB.

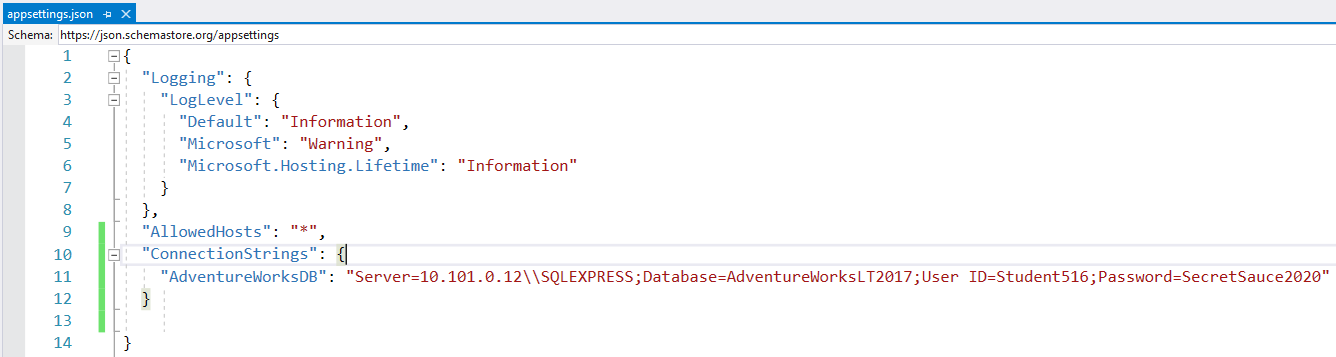
1. Add Connection String in appsettings.json :

"ConnectionStrings": {

"AdventureWorksDB": "Server=10.101.0.12\\SQLEXPRESS;Database=AdventureWorksLT2017;User ID=Student516;Password=SecretSauce2020"

}

The file appsettings.json should look like this:



1. In Startup.cs, add :

public void ConfigureServices(IServiceCollection services)

{

services.AddControllersWithViews(); //already there

services.AddDbContext<AdventureWorksLT2017Context>(options =>

options.UseSqlServer(Configuration.GetConnectionString("AdventureWorksDB")));

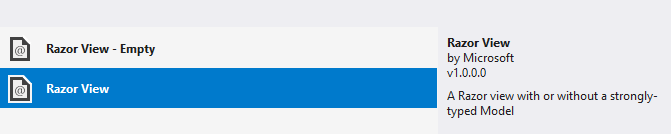
}

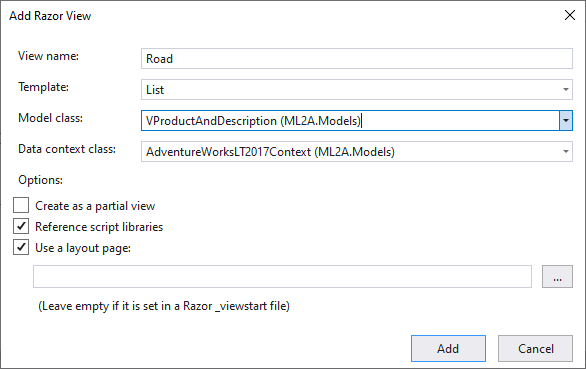
We will have to add these:

using Microsoft.EntityFrameworkCore; //Add this

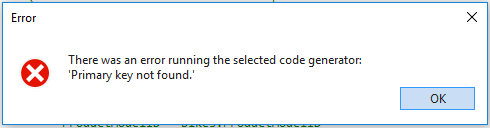
using ML2A.Models; //Add your project name which might be different

1. Run it again and you should see a list of all product categories.
2. Create a Road view: Views 🡪 Bikes 🡪 Right Click 🡪 Add 🡪 View





1. You will get an Error when creating a view based on DB view vProductAndDescription:



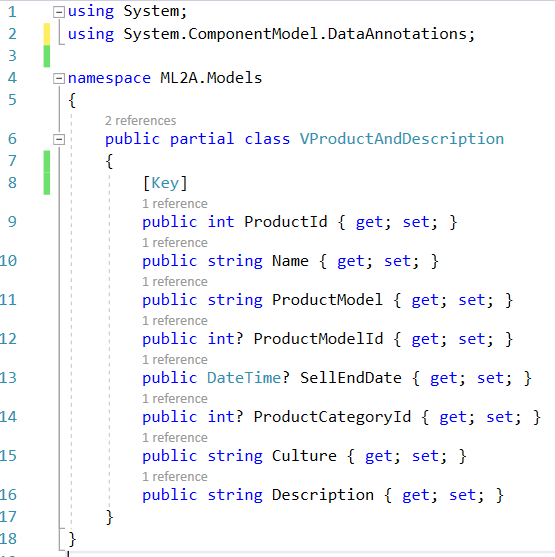
The Entity Framework will automatically create classes from tables but it has trouble converting the database views to classes. When you try to scaffold a class based on a database view, you will receive the above error message.

You manually have to modify the class VProductAndDescription.cs based on the DB view:

Set the view’s primary key:

* Add [Key] data annotation only to ProductId. This makes it primary key.
* Only SellEndDate Property should be nullable. Remove the ? making Properties nullable for all except for SellEndDate.

Save and Build (CTRL+SHIFT+B).



More about Data Annotations:

<https://www.entityframeworktutorial.net/code-first/dataannotation-in-code-first.aspx>

Note: Even with all these modifications, the **scaffolding** **will not work on this DB view**. However, we can query the database view using LINQ and create a custom View.

But there are ways around this… See Milestone 2.