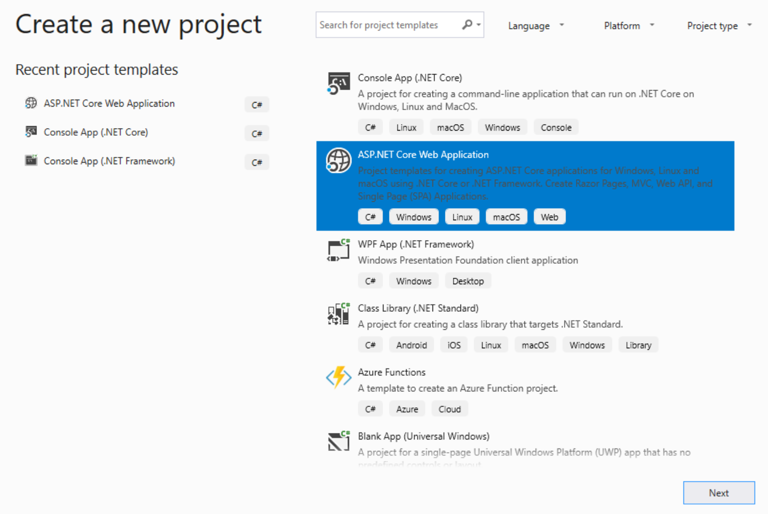
Entity Framework Lab 2

Core Entity Framework Tutorial

This tutorial is taken from `https://docs.microsoft.com/en-us/ef/core/get-started/aspnetcore/new-db?tabs=visual-studio` and `https://github.com/aspnet/EntityFramework.Docs/tree/master/samples/core/GetStarted/AspNetCore/EFGetStarted.AspNetCore.NewDb`.

1 Beginning the lab

1. Open Visual Studio 2019. Select `File` ▶ `New` ▶ `Project`. From the left menu, select `Installed` ▶ `Visual C#` ▶ `.NET Core`. Select `ASP.NET Core Web Application`. Click `Next`. See figure 1.



Console App (.NET Core)
A project for creating a command-line application that can run on .NET Core on Windows, Linux and MacOS.

ASP.NET Core Web Application
Project templates for creating ASP.NET Core applications for Windows, Linux and macOS using .NET Core or .NET Framework. Create Razor Pages, MVC, Web API, and Single Page (SPA) Applications.

WPF App (.NET Framework)
Windows Presentation Foundation client application

Class Library (.NET Standard)
A project for creating a class library that targets .NET Standard.

Azure Functions
A template to create an Azure Function project.

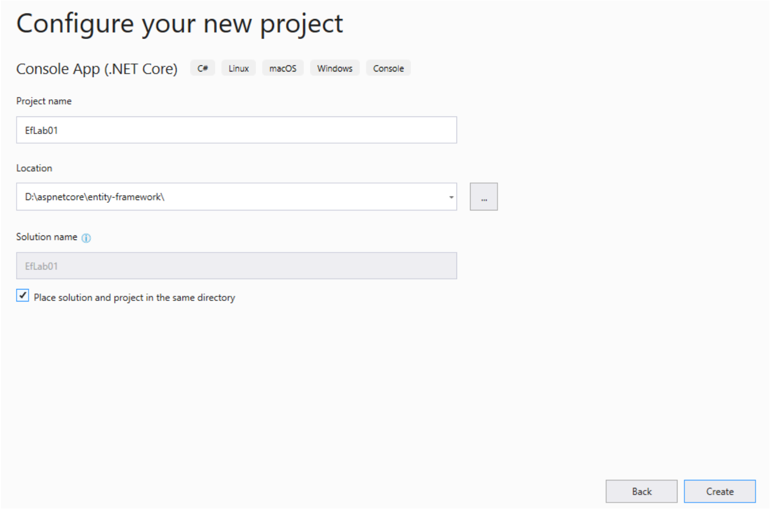
Blank App (Universal Windows)
A project for a single-page Universal Windows Platform (UWP) app that has no predefined controls or layout

Figure 1: Beginning a new ASP.NET App

2. Enter `EfLab02` for the name and click `Create`. See figure 2.

3. In the New ASP.NET Core Web Application dialog: Make sure that .NET Core and ASP.NET Core 2.2 are selected in the drop-down lists. Select the `Web Application (Model-View-Controller)` project template Make sure that Authentication is set to No Authentication Click `Create`. See figure 3.

4. Add a class to the `Models` folder nammed `Class.cs`. Edit the file to match listing 1.



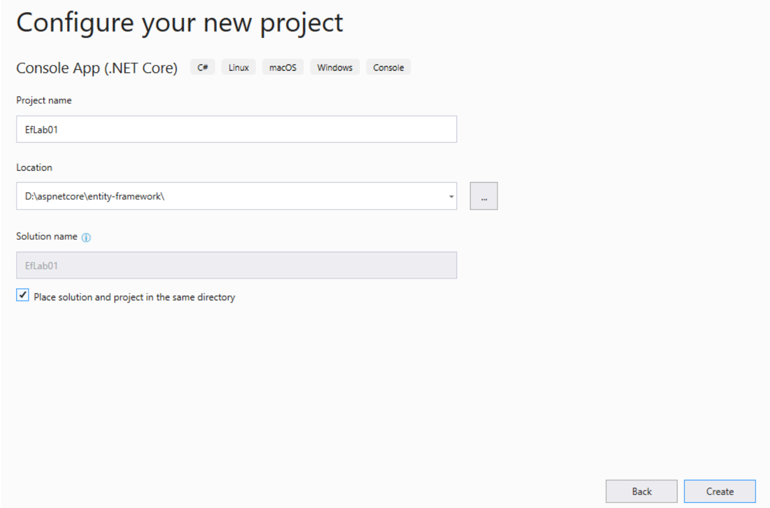
Project name: EfLab01

Location: D:\aspnetcore\entity-framework\

Solution name: EfLab01

☑ Place solution and project in the same directory

Figure 2: Configure a new Console App, first step



Project name: EfLab01

Location: D:\aspnetcore\entity-framework\

Solution name: EfLab01

☑ Place solution and project in the same directory

Figure 3: Configure a new Console App, second step

Listing 1: Contents of `Class.cs`

```
using Microsoft.EntityFrameworkCore;
using System.Collections.Generic;

namespace EfLab02.Models
{
    public class BloggingContext : DbContext
    {
        public BloggingContext(DbContextOptions<BloggingContext> options)
            : base(options)
        { }

        public DbSet<Blog> Blogs { get; set; }
        public DbSet<Post> Posts { get; set; }
    }

    public class Blog
    {
        public int BlogId { get; set; }
        public string Url { get; set; }

        public ICollection<Post> Posts { get; set; }
    }

    public class Post
    {
        public int PostId { get; set; }
        public string Title { get; set; }
        public string Content { get; set; }

        public int BlogId { get; set; }
        public Blog Blog { get; set; }
    }
}
```

5. Add the following `using` statements to the `Startup.cs` file, shown in listing 2.

Listing 2: Add `usig` statements to `Startup.cs`

```
using Microsoft.EntityFrameworkCore;
using EfLab02.Models;
```

6. Add the following statements to the `Startup.cs` file in the `ConfigureServices()` nethod, shown in listing 6.

```
        public void ConfigureServices(IServiceCollection services)
        {
            services.Configure<CookiePolicyOptions>(options =>
            {
                // This lambda determines whether user consent for non-essential cookies is
                    needed for a given request.
                options.CheckConsentNeeded = context => true;
                options.MinimumSameSitePolicy = SameSiteMode.None;
            });

            services.AddMvc().SetCompatibilityVersion(CompatibilityVersion.Version_2_2);

            var connection =
@"Server=(localdb)\mssqllocaldb;Database=EfLab02;Trusted_Connection=True;ConnectRetryCount=0";
            services.AddDbContext<BloggingContext>
                (options => options.UseSqlServer(connection));
            // BloggingContext requires
            // using EFGetStarted.AspNetCore.NewDb.Models;
            // UseSqlServer requires
            // using Microsoft.EntityFrameworkCore;}
```

7. From the Tools menu, select `Tools` ▶ `NuGet Package Manager` ▶ `Package Manager Console` and run the commands shown in listing 3.

Listing 3: Creating the database

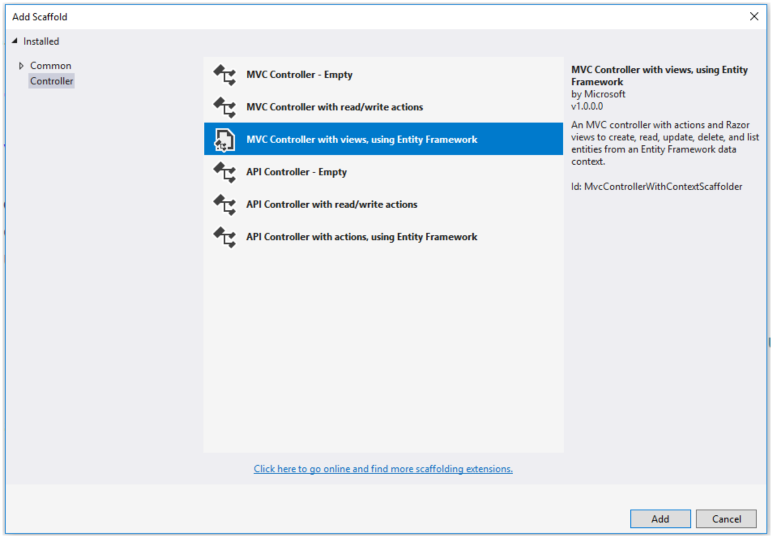
```
Add-Migration InitialCreate
Update-Database
```

This will result in the following output:

```
Each package is licensed to you by its owner. NuGet is not responsible for, nor does it grant any licenses to, third-party packages. Some packages may include dependencies

Package Manager Console Host Version 5.0.0.5917

Type 'get-help NuGet' to see all available NuGet commands.

PM> Add-Migration InitialCreate
Microsoft.EntityFrameworkCore.Infrastructure[10403]
      Entity Framework Core 2.2.2-servicing-10034 initialized 'BloggingContext' using provider 'Microsoft.EntityFrameworkCore.SqlServer' with options: None
To undo this action, use Remove-Migration.
PM> Update-Database
Microsoft.EntityFrameworkCore.Infrastructure[10403]
      Entity Framework Core 2.2.2-servicing-10034 initialized 'BloggingContext' using provider 'Microsoft.EntityFrameworkCore.SqlServer' with options: None
infoverbose: Executed DbCommand (680ms) [Parameters=[], CommandType='Text', CommandTimeout='60']
: Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (680ms) [Parameters=[], CommandType='Text', CommandTimeout='60']
      CREATE DATABASE [EfLab02];
Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (90ms) [Parameters=[], CommandType='Text', CommandTimeout='60']
      IF SERVERPROPERTY('EngineEdition') <> 5
      BEGIN
          ALTER DATABASE [EfLab02] SET READ_COMMITTED_SNAPSHOT ON;
      END;
Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (10ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      CREATE TABLE [__EFMigrationsHistory] (
          [MigrationId] nvarchar(150) NOT NULL,
          [ProductVersion] nvarchar(32) NOT NULL,
          CONSTRAINT [PK___EFMigrationsHistory] PRIMARY KEY ([MigrationId])
      );
Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (10ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      SELECT OBJECT_ID(N'[__EFMigrationsHistory]');
Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (5ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      SELECT [MigrationId], [ProductVersion]
      FROM [__EFMigrationsHistory]
      ORDER BY [MigrationId];
Applying migration '20190509152855_InitialCreate'.
Microsoft.EntityFrameworkCore.Migrations[20402]
      Applying migration '20190509152855_InitialCreate'.
Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (4ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      CREATE TABLE [Blogs] (
          [BlogId] int NOT NULL IDENTITY,
          [Url] nvarchar(max) NULL,
          CONSTRAINT [PK_Blogs] PRIMARY KEY ([BlogId])
      );
Microsoft.EntityFrameworkCore.Database.Command[20101]
```

```
      Executed DbCommand (2ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      CREATE TABLE [Posts] (
          [PostId] int NOT NULL IDENTITY,
          [Title] nvarchar(max) NULL,
          [Content] nvarchar(max) NULL,
          [BlogId] int NOT NULL,
          CONSTRAINT [PK_Posts] PRIMARY KEY ([PostId]),
          CONSTRAINT [FK_Posts_Blogs_BlogId] FOREIGN KEY ([BlogId]) REFERENCES [Blogs] ([BlogId]) ON DELETE CASCADE
      );
Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (1ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      CREATE INDEX [IX_Posts_BlogId] ON [Posts] ([BlogId]);
Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (3ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      INSERT INTO [__EFMigrationsHistory] ([MigrationId], [ProductVersion])
      VALUES (N'20190509152855_InitialCreate', N'2.2.2-servicing-10034');
Done.
PM>
```

8. Right-click on the Controllers folder in Solution Explorer and select `Add` ▶ `Controller`. See figure 4.


Add Scaffold
Installed
Common
Controller
MVC Controller - Empty
MVC Controller with read/write actions
MVC Controller with views, using Entity Framework
API Controller - Empty
API Controller with read/write actions
API Controller with actions, using Entity Framework
Click here to go online and find more scaffolding extensions.
Add
Cancel

MVC Controller with views, using Entity Framework
by Microsoft
v1.0.0.0

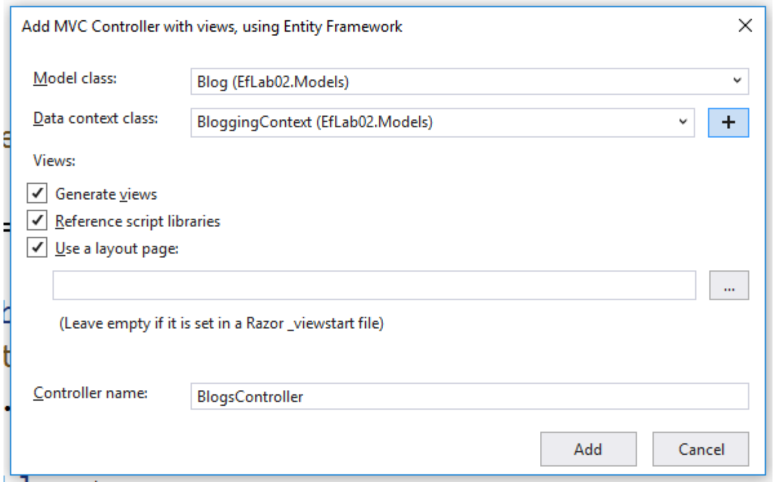
An MVC controller with actions and Razor views to create, read, update, delete, and list entities from an Entity Framework data context.

Id: MvcControllerWithContextScaffolder

Figure 4: Adding a controller with views.

9. Select *MVC Controller with views, using Entity Framework* and click `Add`. Set Model class to `Blog` and Data context class to `BloggingContext`. Click `Add`. See figure 5.

10. Select `Debug` ▶ `Start Without Debugging`. Navigate to `PathTo/Blogs`. Use the `Create New` link to create some blog entries. See figure 6.



Add MVC Controller with views, using Entity Framework

Model class: Blog (EfLab02.Models)

Data context class: BloggingContext (EfLab02.Models)

Views:

☑ Generate views

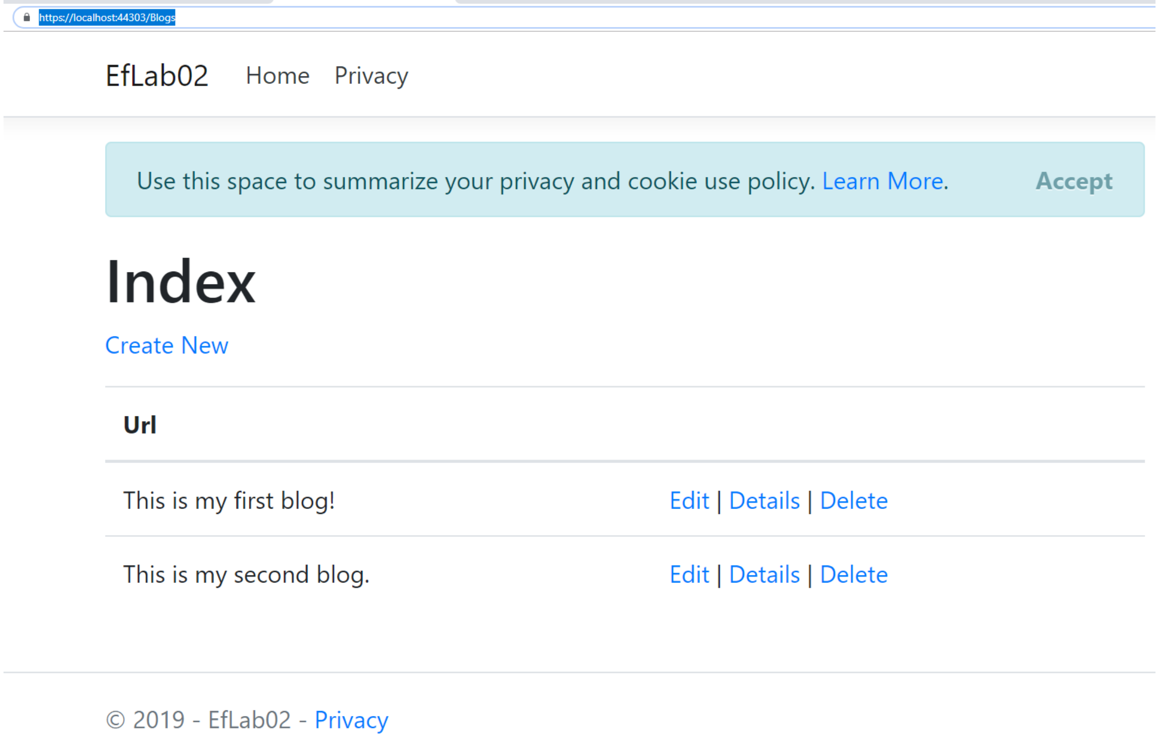
☑ Reference script libraries

☑ Use a layout page:

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

Controller name: BlogsController

Figure 5:



https://localhost:44303/Blogs

Use this space to summarize your privacy and cookie use policy. Learn More.

Index

Create New

Url	
This is my first blog!	Edit | Details | Delete
This is my second blog.	Edit | Details | Delete

© 2019 - EfLab02 - Privacy

Figure 6: Running appication