# Creating a .NET Core MVC Project with Code First Entity Framework

## Create a .NET Core MVC Project

* File, New Project, Web, ASP.NET Core Web Application (or search in VS2019)
* Select MVC

## Add your EF models and DbContext.

* Add your entity model classes. Plain C# classes. (Product, Customer, Vendor, etc.)
* Add your database context class and add a constructor. (code below assumes the context class is named “MyDB”)

public class MyDB : DbContext

{

public DbSet<Product> Products { get; set; }

public DbSet<Customer> Customers { get; set; }

// constructor to allow .NET Core to do dependency injection

**public MyDB(DbContextOptions options) : base(options)**

**{ }**

}

## Edit Startup.cs:

* Add a using statement: using Microsoft.EntityFrameworkCore;
* In the ConfigureServices method add your connection string.

services.AddMvc().SetCompatibilityVersion(CompatibilityVersion.Version\_2\_2);

**var connection = @"Data Source=(localdb)\MSSQLLocalDB;Initial Catalog=DotNetCoreMVC.Models.MyDB;Integrated Security=True;Connect Timeout=30;Encrypt=False;TrustServerCertificate=True;ApplicationIntent=ReadWrite;MultiSubnetFailover=False";**

**services.AddDbContext<Models.MyDB>**

**(options => options.UseSqlServer(connection));**

Note: Storing the connection string in the code is BAD! Store it in appsettings.json and retrieve with:  
 var connetion = Configuration["yourConnectionStringName"];

Note: If you copied your connection string from a “server” connection property, add “Initial Catalog=youNewDatabaseName;” to the connection string.

Build the project

## Configure migrations:

Tools > NuGet Package Manager > Package Manager Console

Add-Migration InitialCreate  
Update-Database

## Add your Controllers and Views

Note: The new database will not be created until the first reference to an EF object, typically in a controller.