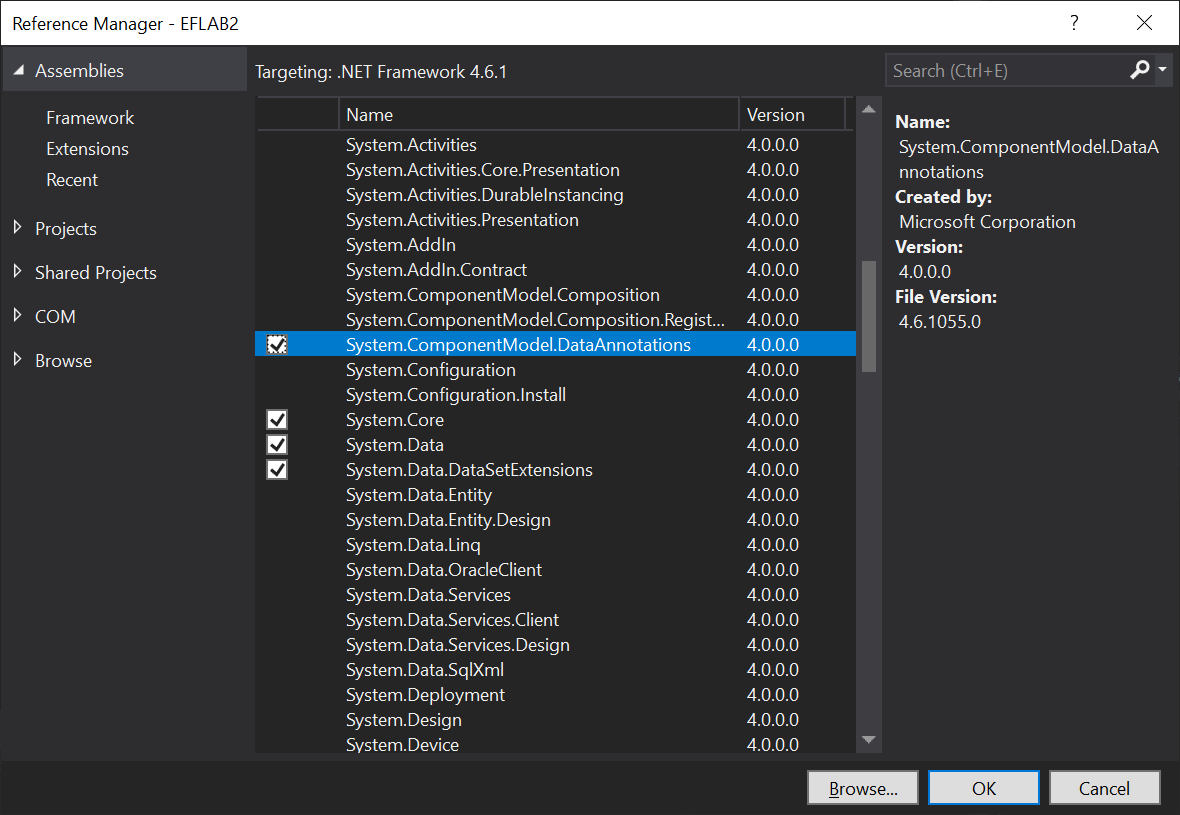
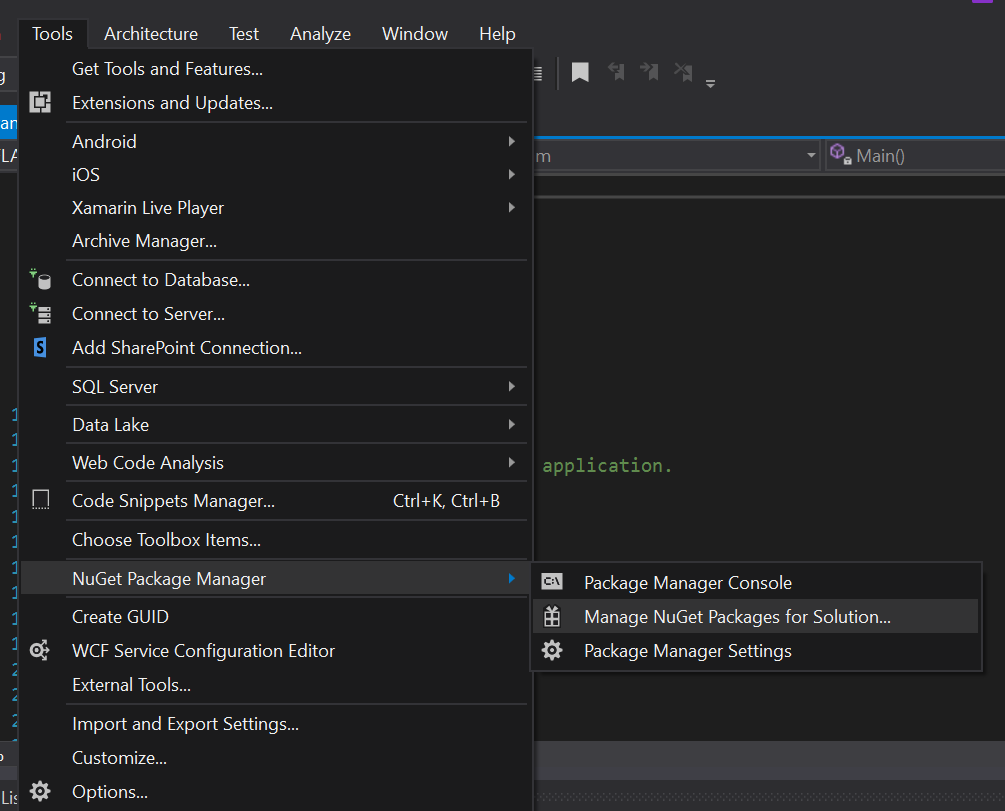
Week 10 Part 1 Second Part – Entity Framework setup and configure, CODE FIRST

You should already have a project which the annotated classes from part one. The order is not so important it is possible that you set up and configure EF with a DB before creating the entity classes. However the assumption is that you already have the entity classes you are going to persist and that you completed Part 1

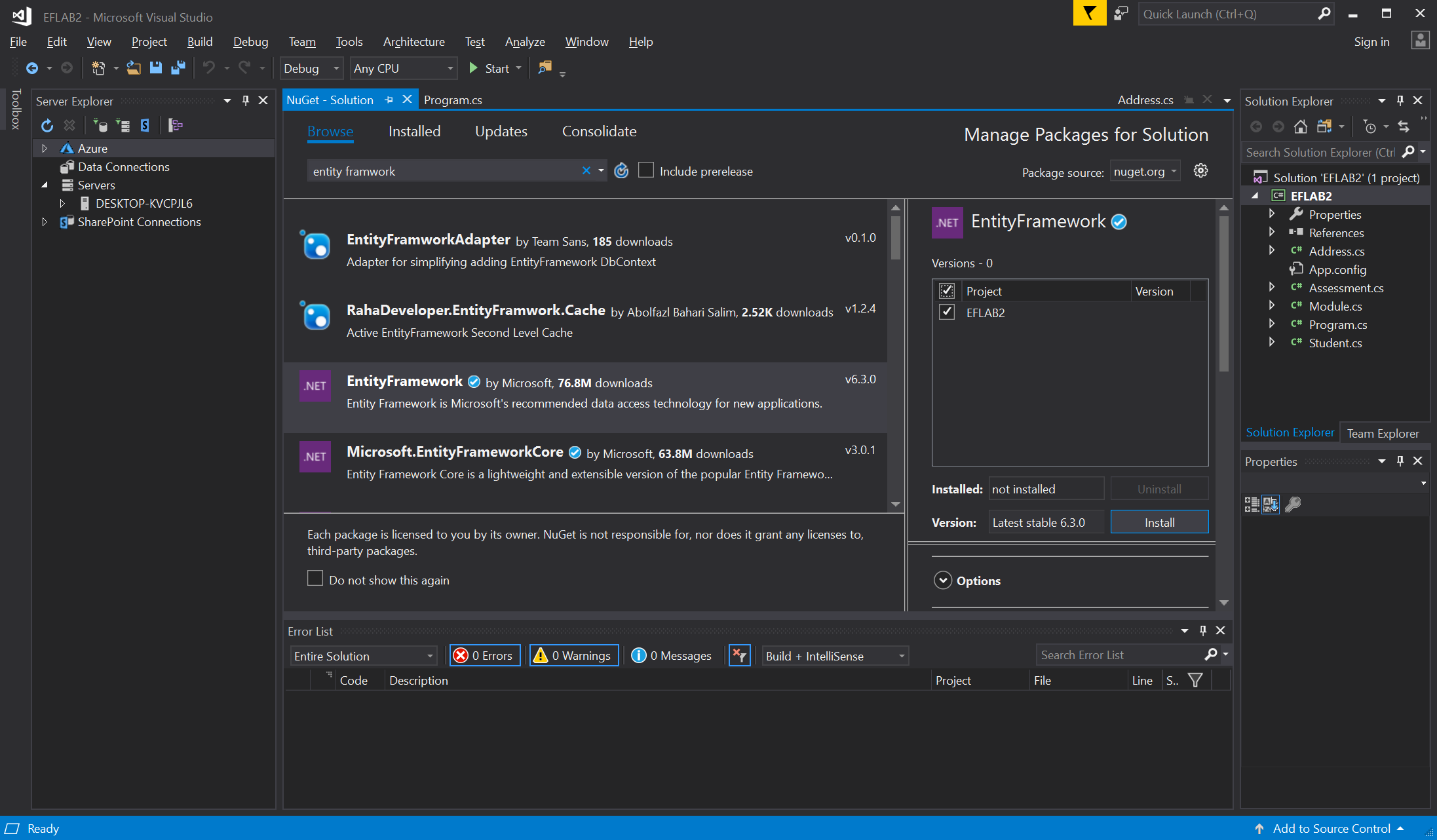
Note that I am rebuilding my app as not a form based app but a console one and it missed the DataAnnotation assembly, so I had to add it manually.



Open the Manage MuGet Packages for Solution…



Select Browse and put in the search Entity Framework



Click and then select the project to the right hand side.

Then click install, and agree any further dialogs to get it to install.

You should check that it has installed correctly the App.conig file will have changed to show the dependency, and allow you to validate that it has been installed correctly. We are going to do some more work with this file as it is needed to link the DB file to the project.

<?xml version="1.0" encoding="utf-8"?>

<configuration>

<configSections>

<!-- For more information on Entity Framework configuration, visit http://go.microsoft.com/fwlink/?LinkID=237468 -->

**<section name="entityFramework"**

**type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"**

**requirePermission="false"/>**

</configSections>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.6.1"/>

</startup>

<entityFramework>

<providers>

**<provider invariantName="System.Data.SqlClient" type="System.Data.Entity.SqlServer.SqlProviderServices, EntityFramework.SqlServer"/>**

</providers>

</entityFramework>

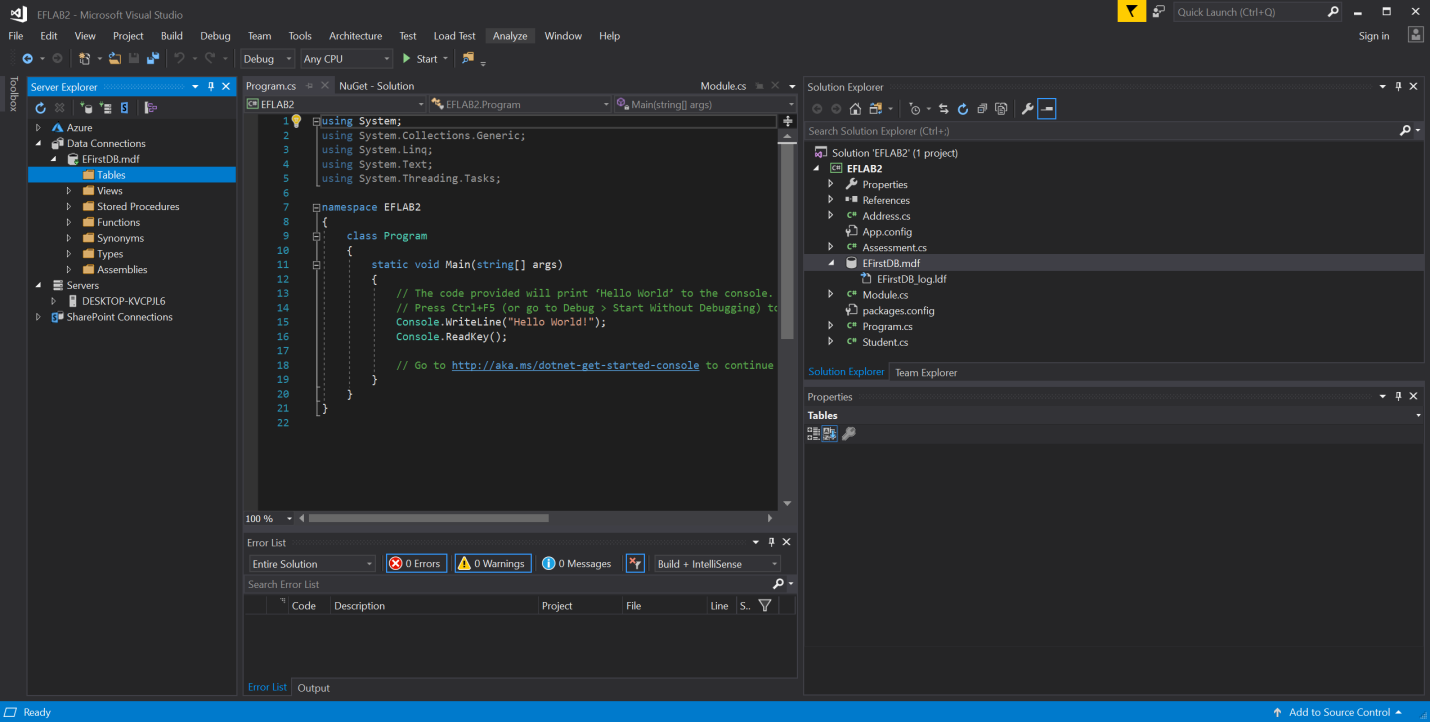
</configuration>

Next create an EMPTY database. We are *not* going to fill it with data.

In the solutions explorer, add new item for the project and select Server–Based Database

Name the file EFirstDB.mdf

Click on the EFirstDB file in the explorer and you will see there are no tables as EF is still not fully configured.



Next we need to manually alter the App.config to add in the database connection.

Add the following section to the App.config it needs to be encapsulated (surrounded by) <connectionStrings> to </connectionStrings> note that you DB location will be a little different to the one shown as it will depend on your specific connection string which can be found in the database properties.

<?xml version="1.0" encoding="utf-8"?>

<configuration>

<configSections>

<!-- For more information on Entity Framework configuration, visit http://go.microsoft.com/fwlink/?LinkID=237468 -->

<section name="entityFramework"

type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"

requirePermission="false"/>

</configSections>

**<connectionStrings>**

**<add name="conString" connectionString="Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=E:\EFLAB2\EFLAB2\EFirstDB.mdf;Integrated Security=True"**

**providerName="System.Data.SqlClient" />**

**</connectionStrings>**

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.6.1"/>

</startup>

<entityFramework>

<providers>

<provider invariantName="System.Data.SqlClient" type="System.Data.Entity.SqlServer.SqlProviderServices, EntityFramework.SqlServer"/>

</providers>

</entityFramework>

</configuration>

The ‘name’ on the connection string is what is needed in the DBcontect class,

App.config has  **<add name="conString"** is and the dbcontext has the following again with **base("name=conString")**

class MyDBEntities : DbContext

{

public MyDBEntities()

: **base("name=conString")**

{

// Database.SetInitializer(new MigrateDatabaseToLatestVersion<MyDBEntities);

}

public DbSet<Student> Students { get; set; }

public DbSet<Module> Modules { get; set; }

public DbSet<Address> Addresses { get; set; }

public DbSet<Assessment> Assessments { get; set; }

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

base.OnModelCreating(modelBuilder);

}

}

Note that I commented out the following line, this is in the documentation as is also given as examples however there are lots of posts online saying to comment it out!

// Database.SetInitializer(new MigrateDatabaseToLatestVersion<MyDBEntities

Open NuGet Package manager console and type

enable-migrations –EnableAutomaticMigration:$true

you should get no errors of warnings the following is what you should get

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

PM> enable-migrations –EnableAutomaticMigration:$true

Checking if the context targets an existing database...

PM>

There are still not DB tables created we must use the context first.

using (var context = new UniContext())

{

var studentList = context.Students.ToList<Student>();

Address a = new Address();

a.AddressID = 1;

a.NumbnerOrName = "20";

a.PostCode = "LL00 GP4";

a.Street = "BackOfBehond";

a.Town = "Trumpton";

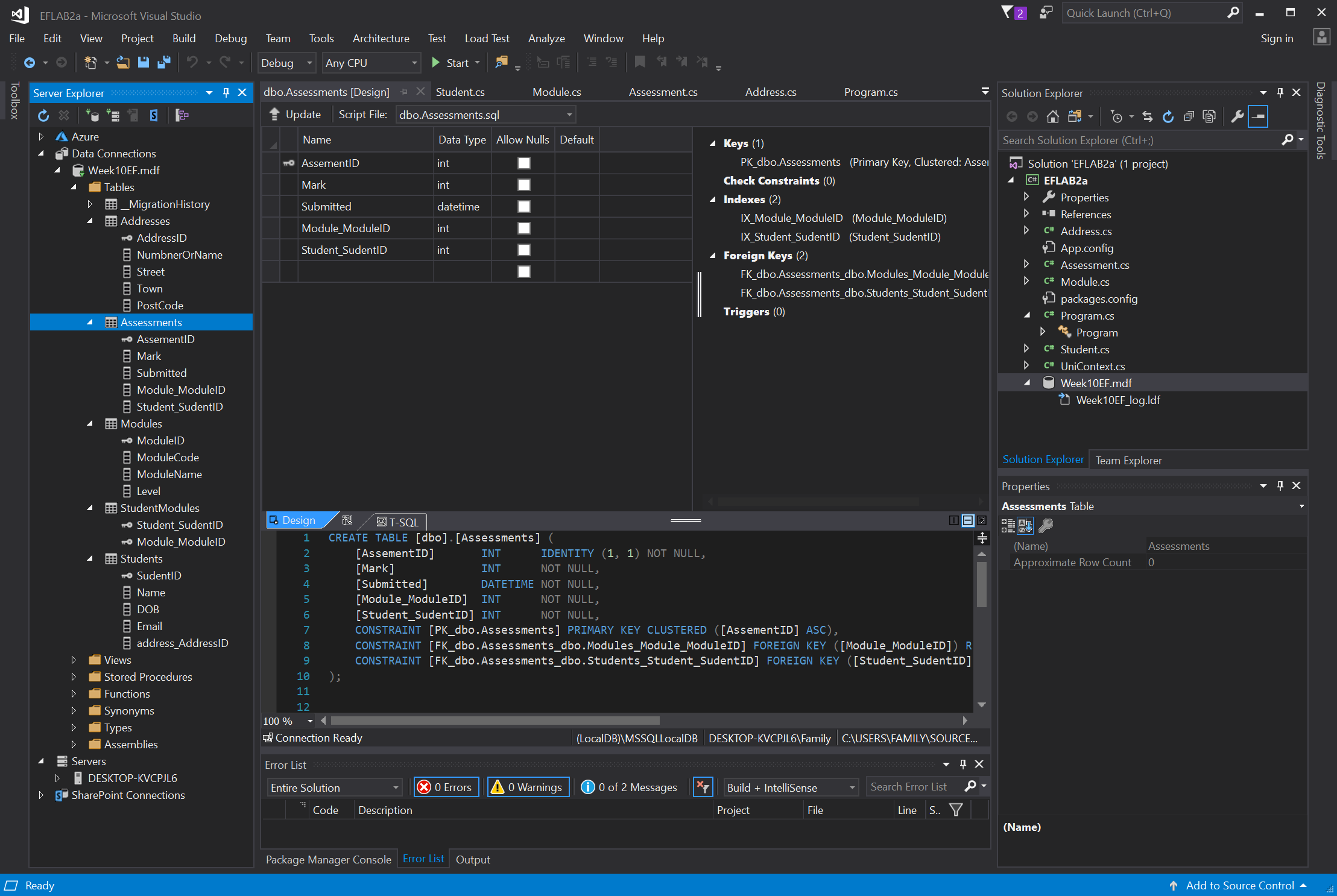
context.Addresses.Add(a);

context.SaveChanges();

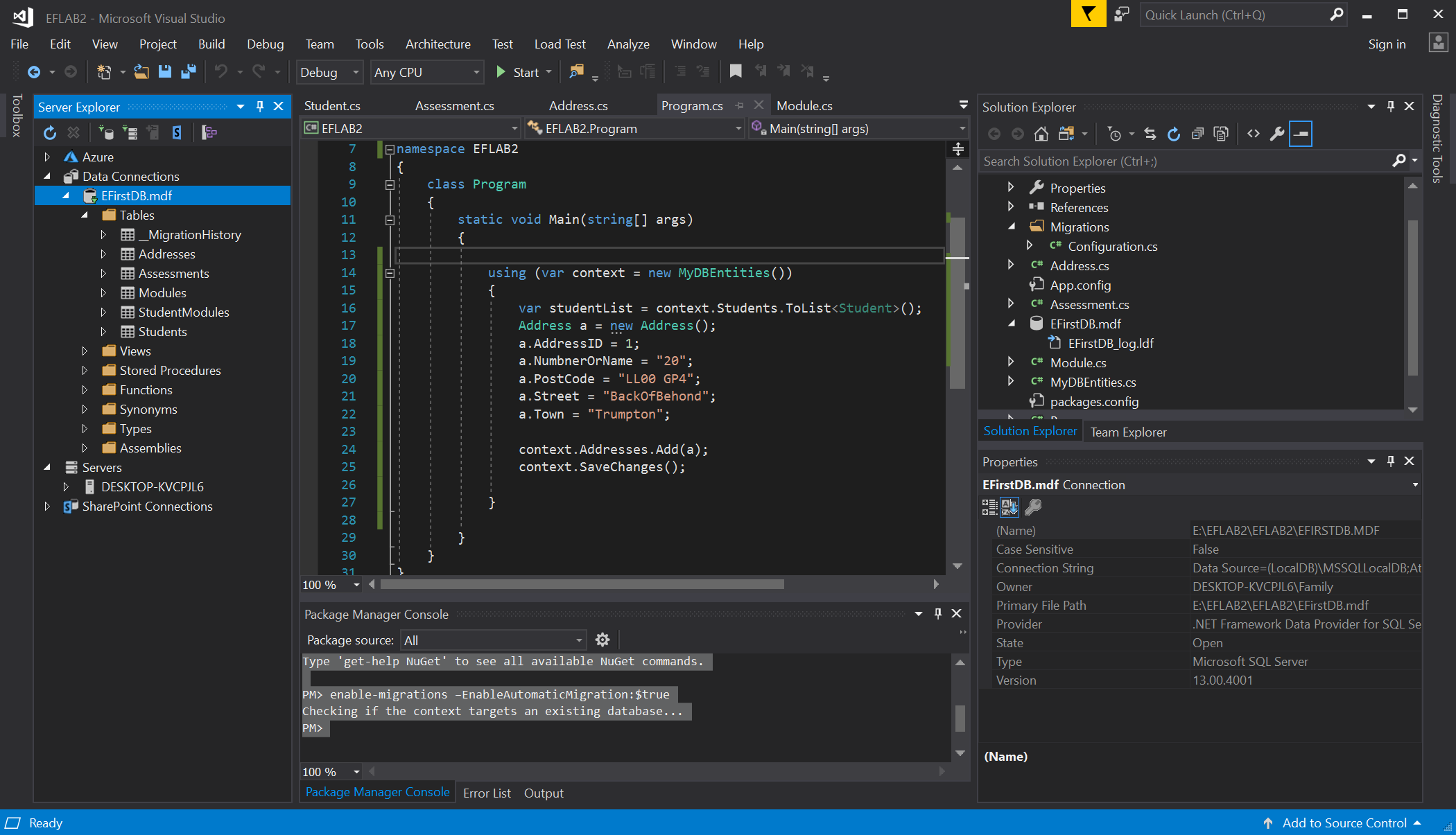
}

}

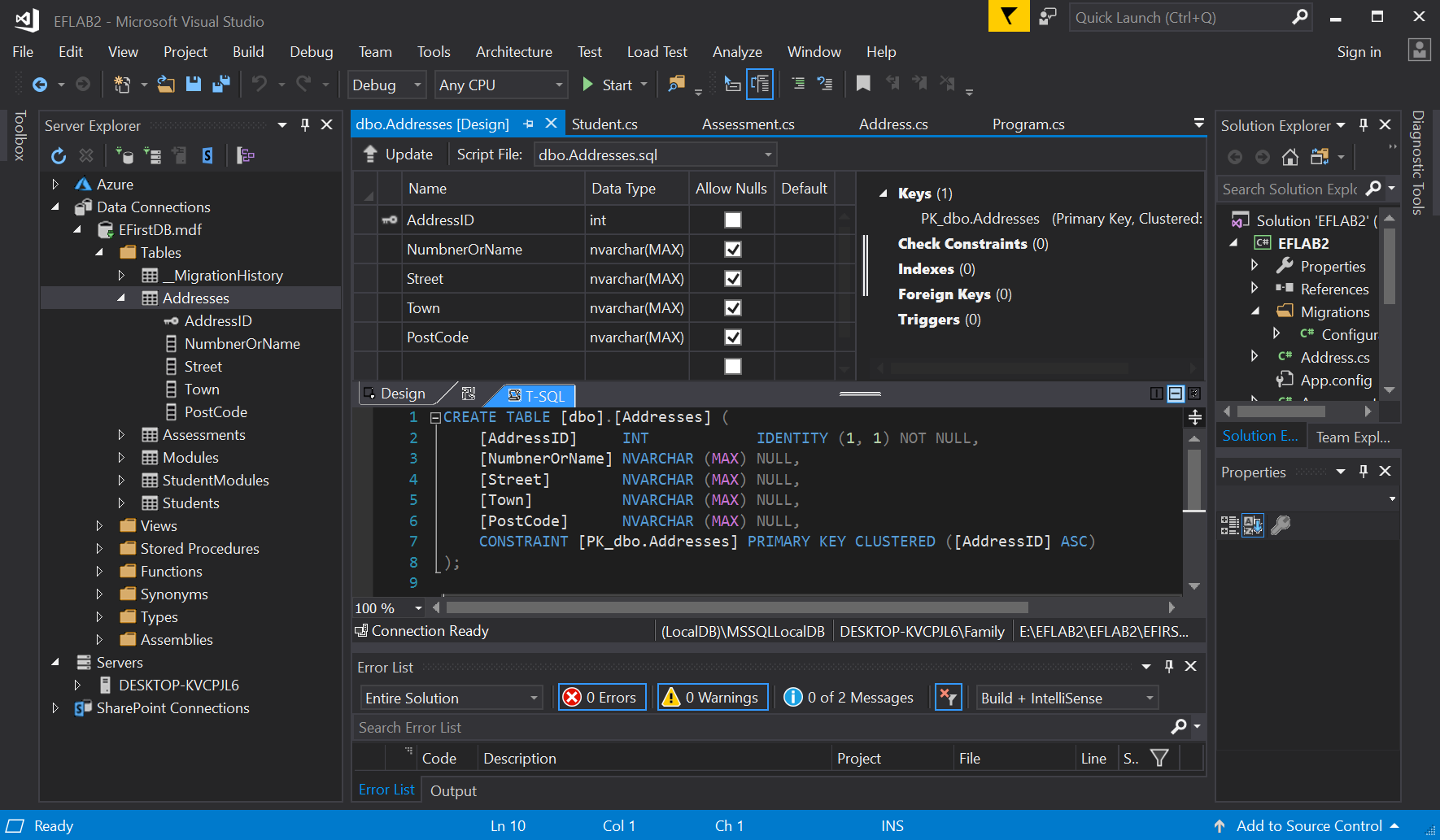
If you refresh the view of the database then you should see the tables



This was the code used to insert a record into the DB



The table which is having a record inserted into to, and we can see he SQL which was used to create the tables all made by EF.



The actual record on the DB

