**Създаване на нов проект**

1. Правя ново Repository в GitHub и го клонирам на диска в целевата папка
2. В целевата папка си правя папки Documentation и Source, в която ще си сложа кода.
3. Отварям Visual Studio и в папка Source си правя Blank Solution, като му задавам подходящо име.
4. В Solution-а си правя папки: Data, Logic, Web
5. В папка Data си правя 2 нови проекта от тип Class Library, с имена **solutionName.Data** и **solutionName.Data.Models** и трия генерираните класове.
6. В папка Web ви правя нов ASP.NET Web Application с име **solutionName.Web**
7. На проекта **solutionName.Web**, десен клик и избирам Set as StartUp Project
8. Проба, дали работи
9. Ъпдейтвам всички NuGet Пакети и рестартирам VS
10. Променям темата(ако желая)

* Инсталирам желаната тема от NuGet
* В BundleConfig променям bootstrap.css на bootstrap.cerulean.css
* Пак там зачиствам Modernizr и го деинсталирам от NuGet пакетите.
* Във View-то \_Layout.cshtml зачиствам реда за Modernizr и в първият ред на body-то променям класа от navbar-inverse на navbar-default

1. Ще премахна файла **IdentityModels.cs**, като класовете **ApplicationUser** ще изнеса в **solutionName.Data.Models,** а **ApplicationDbContext** в **solutionName.Data,** като ще ги реферирам преди това в **solutionName.Web** . За целта – десен клик на **solutionName.Web 🡪 References 🡪 Add Reference… 🡪** слагам отметките пред **solutionName.Data** и **solutionName.Data.Models.**
2. Десен клик на **solutionName.Data**  **🡪 References 🡪 Add Reference… 🡪** слагам отметките пред **solutionName.Data.Models.**
3. Правя си нов клас ApplicationUser в **solutionName.Data.Models** в който копирам съдържанието на класа от файла **IdentityModels.cs**
4. Правя си нов клас ApplicationDbContext в **solutionName.Data,** в който копирам съдържанието на класа от файла **IdentityModels.cs**
5. Копирам using-ите от **IdentityModels.cs** в двата класа и го изтривам.
6. Десен клик на Solution-a 🡪 Manage NuGet Packages 🡪 намирам **Microsoft.AspNet.Identity.EntityFramework** 🡪 слагам отметките пред **solutionName.Data** и **solutionName.Data.Models,** за да се инсталира в тях
7. Слагам необходимите using-и и махам ненужните.
8. Build, за да ми покаже къде трябва да си оправя референциите
9. Оправям ги и пак Rebuild и всичко е Ок.
10. Променям ConnectionString-a да е

<connectionStrings>

<add name="DefaultConnection"

connectionString="Data Source=JDIMOV\SQLEXPRESS;

Initial Catalog=SchoolContext;

Integrated Security=True"

providerName="System.Data.SqlClient" />

</connectionStrings>

1. Пускам, регистрирам потребител – и работи!
2. За проба в папка **Data**, в проекта **solutionName.Data.Models** си правя клас **Student**.

using System;

using System.Collections.Generic;

namespace ContosoUniversityMyVersion.Model

{

public class Student

{

public int ID { get; set; }

public string LastName { get; set; }

public string FirstMidName { get; set; }

public DateTime EnrollmentDate { get; set; }

public virtual ICollection<Enrollment> Enrollments { get; set; }

}

}

1. Добавям и класовете(моделите):

**Enrollment.cs**

namespace ContosoUniversityMyVersion.Model

{

public enum Grade

{

A, B, C, D, F

}

public class Enrollment

{

public int EnrollmentID { get; set; }

public int CourseID { get; set; }

public int StudentID { get; set; }

public Grade? Grade { get; set; }

public virtual Course Course { get; set; }

public virtual Student Student { get; set; }

}

}

**Course.cs**

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations.Schema;

namespace ContosoUniversityMyVersion.Model

{

public class Course

{

[DatabaseGenerated(DatabaseGeneratedOption.None)]

public int CourseID { get; set; }

public string Title { get; set; }

public int Credits { get; set; }

public virtual ICollection<Enrollment> Enrollments { get; set; }

}

}

1. Идеята е, този **Student , Enrollment и Course,** да ги добавя в списъка с DbSet-ове в контекста, за да видя как изглежда в базата и да видя, дали всичко го има в базата данни.

Правя си DatabaseContext, като за целта в **solutionName.Data** добавям нов клас **SchoolContext.cs** със следното съдържание

**SchoolContext.cs**

using ContosoUniversityMyVersion.Model;

using System.Data.Entity;

using System.Data.Entity.ModelConfiguration.Conventions;

namespace ContosoUniversityMyVersion.Data

{

public class SchoolContext : DbContext

{

public SchoolContext() : base("SchoolContext")

{

}

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

public DbSet<Enrollment> Enrollments { get; set; }

public DbSet<Course> Courses { get; set; }

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

modelBuilder.Conventions.Remove<PluralizingTableNameConvention>();

}

}

}

1. В папка **solutionName.Data** създавам нов файл с име **SchoolInitializer.cs**, в който се намира Seed метод, които автометично след създаването на базата данни, ще попълва подадените в него данни за тестове.

**SchoolInitializer.cs**

using System;

using System.Collections.Generic;

using ContosoUniversityMyVersion.Model;

namespace ContosoUniversityMyVersion.Data

{

public class SchoolInitializer : System.Data.Entity.DropCreateDatabaseIfModelChanges<SchoolContext>

{

protected override void Seed(SchoolContext context)

{

var students = new List<Student>

{

new Student{FirstMidName="Carson",LastName="Alexander",EnrollmentDate=DateTime.Parse("2005-09-01")},

new Student{FirstMidName="Meredith",LastName="Alonso",EnrollmentDate=DateTime.Parse("2002-09-01")},

new Student{FirstMidName="Arturo",LastName="Anand",EnrollmentDate=DateTime.Parse("2003-09-01")},

new Student{FirstMidName="Gytis",LastName="Barzdukas",EnrollmentDate=DateTime.Parse("2002-09-01")},

new Student{FirstMidName="Yan",LastName="Li",EnrollmentDate=DateTime.Parse("2002-09-01")},

new Student{FirstMidName="Peggy",LastName="Justice",EnrollmentDate=DateTime.Parse("2001-09-01")},

new Student{FirstMidName="Laura",LastName="Norman",EnrollmentDate=DateTime.Parse("2003-09-01")},

new Student{FirstMidName="Nino",LastName="Olivetto",EnrollmentDate=DateTime.Parse("2005-09-01")}

};

students.ForEach(s => context.Students.Add(s));

context.SaveChanges();

var courses = new List<Course>

{

new Course{CourseID=1050,Title="Chemistry",Credits=3,},

new Course{CourseID=4022,Title="Microeconomics",Credits=3,},

new Course{CourseID=4041,Title="Macroeconomics",Credits=3,},

new Course{CourseID=1045,Title="Calculus",Credits=4,},

new Course{CourseID=3141,Title="Trigonometry",Credits=4,},

new Course{CourseID=2021,Title="Composition",Credits=3,},

new Course{CourseID=2042,Title="Literature",Credits=4,}

};

courses.ForEach(s => context.Courses.Add(s));

context.SaveChanges();

var enrollments = new List<Enrollment>

{

new Enrollment{StudentID=1,CourseID=1050,Grade=Grade.A},

new Enrollment{StudentID=1,CourseID=4022,Grade=Grade.C},

new Enrollment{StudentID=1,CourseID=4041,Grade=Grade.B},

new Enrollment{StudentID=2,CourseID=1045,Grade=Grade.B},

new Enrollment{StudentID=2,CourseID=3141,Grade=Grade.F},

new Enrollment{StudentID=2,CourseID=2021,Grade=Grade.F},

new Enrollment{StudentID=3,CourseID=1050},

new Enrollment{StudentID=4,CourseID=1050,},

new Enrollment{StudentID=4,CourseID=4022,Grade=Grade.F},

new Enrollment{StudentID=5,CourseID=4041,Grade=Grade.C},

new Enrollment{StudentID=6,CourseID=1045},

new Enrollment{StudentID=7,CourseID=3141,Grade=Grade.A},

};

enrollments.ForEach(s => context.Enrollments.Add(s));

context.SaveChanges();

}

}

}

Поради факта, че е зададено DropCreateDatabaseIfModelChanges<SchoolContext>, ако направим промени в базата, тя автоматично ще се ъпдейтне.(Да, ама Не!)

1. За да може EntityFramework да използва нашия  initializer class, добавяме в entityFramework елемента, който се намира във Web.Config файла следния код:

<entityFramework>

<contexts>

<context type="ContosoUniversityMyVersion.Data.SchoolContext,

ContosoUniversityMyVersion.Data">

<databaseInitializer

type="ContosoUniversityMyVersion.Data.SchoolInitializer,

ContosoUniversityMyVersion.Data" />

</context>

</contexts>

<defaultConnectionFactory......

1. Създавам StudentController и View-та чрез scaffolding
2. Проверка – Работи!!!
3. Добавям поле EmailAddress в класа Student, допълвам в контролера и view-тата, трия базата, пускам и работи