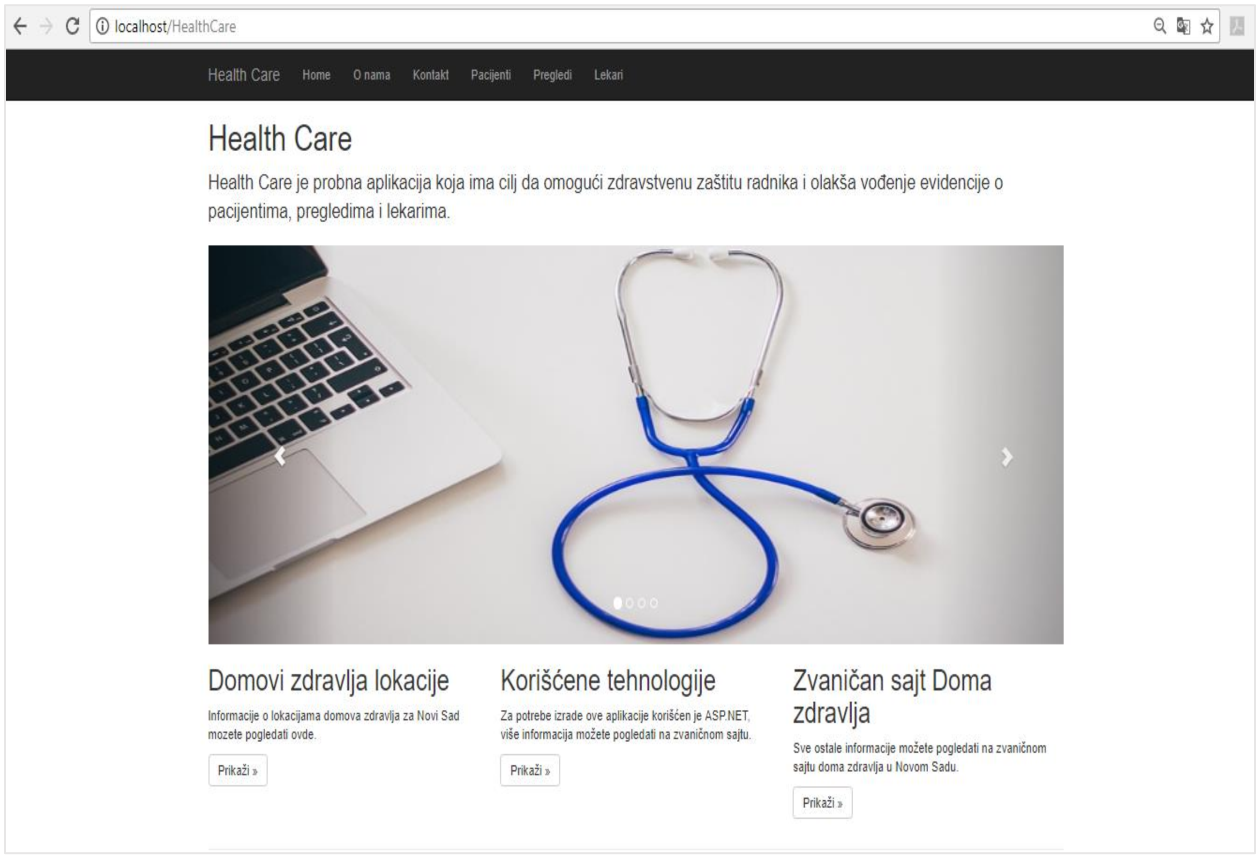
Introduction

This project shows us how to create ASP.NET MVC 5 web application and implement Object Relation Mapping using Entity Framework and Visual Studio.

In this project we will build a web site for Health Care Center. Health Care application contains functionalities such as view details about patient and update patient, medical examination and doctor information, create new patient and doctor, create new medical examination, delete patient, upload and edit patient documentation, download files about medical examinations, upload and edit doctor photo.



Health Care

Health Care je probna aplikacija koja ima cilj da omogući zdravstvenu zaštitu radnika i olakša vođenje evidencije o pacijentima, pregledima i lekarima.

Domovi zdravlja lokacije

Informacije o lokacijama domova zdravlja za Novi Sad mozete pogledati ovde.

Korišćene tehnologije

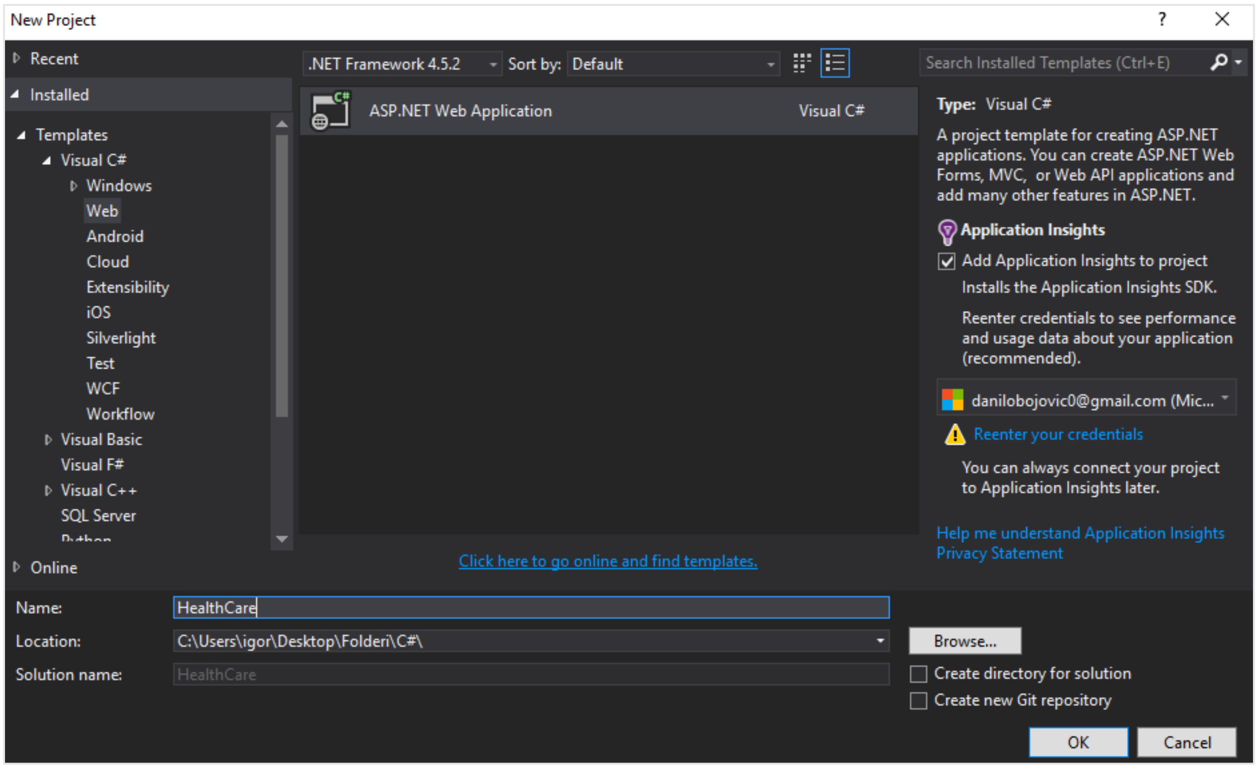
Za potrebe izrade ove aplikacije korišćen je ASP.NET, više informacija možete pogledati na zvaničnom sajtu.

Zvaničan sajt Doma zdravlja

Sve ostale informacije možete pogledati na zvaničnom sajtu doma zdravlja u Novom Sadu.

Creating HealthCare ASP.NET MVC Application

After opening Visual Studio, click New on the File menu, then select Project -> Web -> ASP.NET Web Application.



Type: Visual C#

A project template for creating ASP.NET applications. You can create ASP.NET Web Forms, MVC, or Web API applications and add many other features in ASP.NET.

Application Insights

☑ Add Application Insights to project

Installs the Application Insights SDK.

Reenter credentials to see performance and usage data about your application (recommended).

danilobojovic0@gmail.com (Mic...

Reenter your credentials

You can always connect your project to Application Insights later.

Help me understand Application Insights

Privacy Statement

Name: HealthCare

Location: C:\Users\igor\Desktop\Folderi\C#\

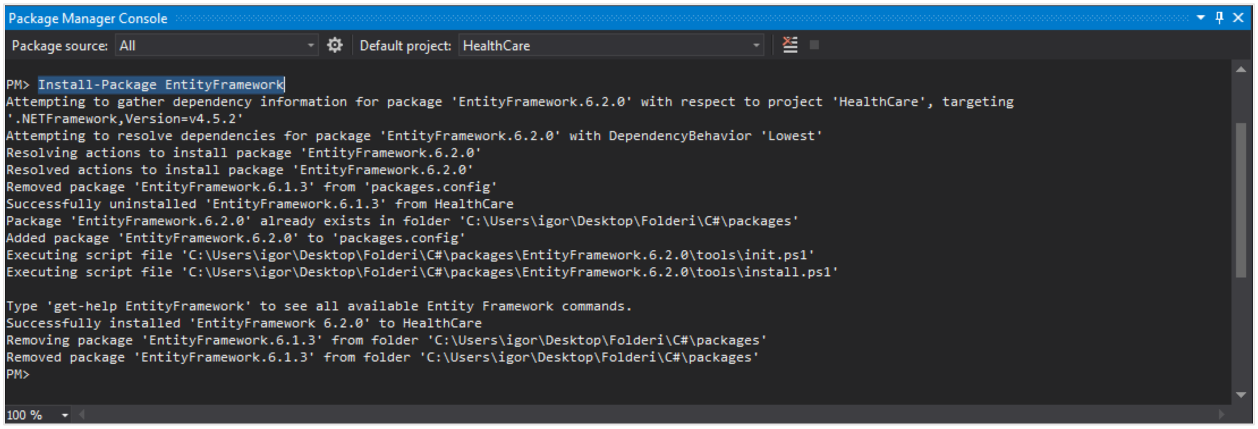
Solution name: HealthCare

☐ Create directory for solution

☐ Create new Git repository

Picture 1: Creating a new project named HealthCare

From the tools menu choose Package Manager Console and enter `Install-Package EntityFramework` command.

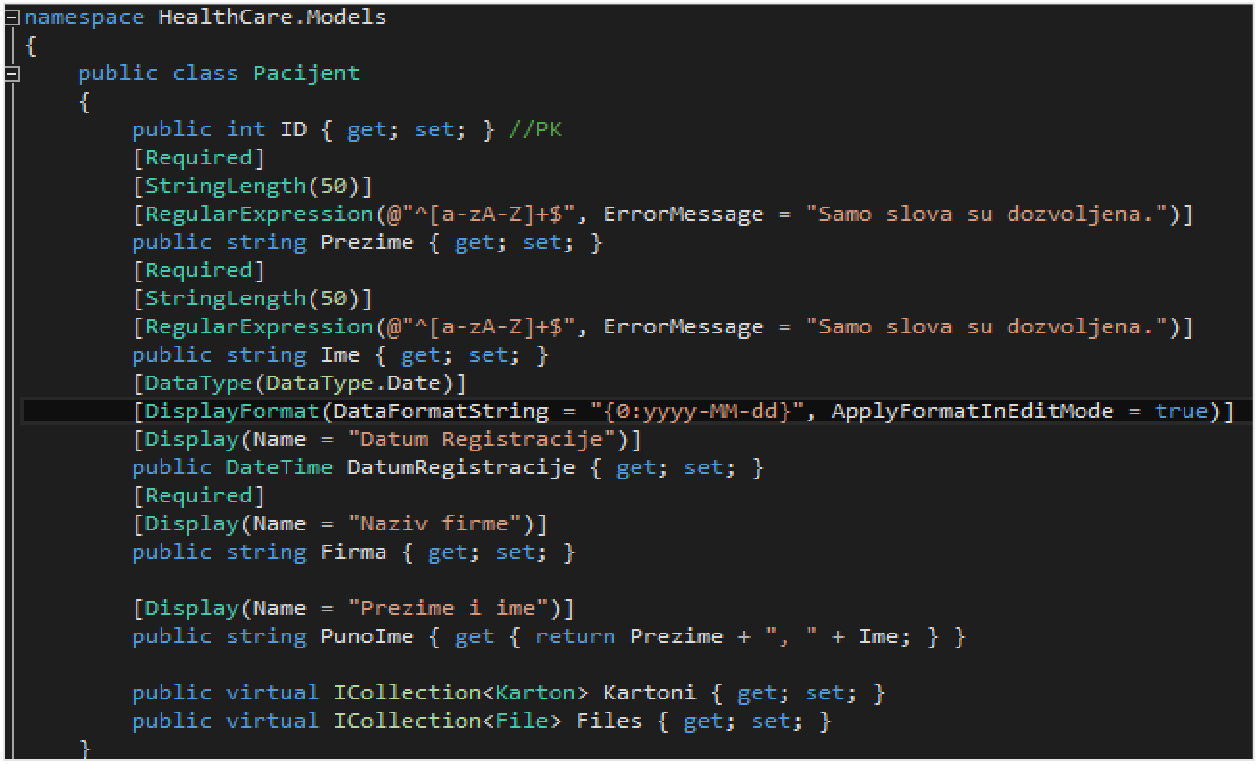


```
PM> Install-Package EntityFramework
Attempting to gather dependency information for package 'EntityFramework.6.2.0' with respect to project 'HealthCare', targeting
'.NETFramework,Version=v4.5.2'
Attempting to resolve dependencies for package 'EntityFramework.6.2.0' with DependencyBehavior 'Lowest'
Resolving actions to install package 'EntityFramework.6.2.0'
Resolved actions to install package 'EntityFramework.6.2.0'
Removed package 'EntityFramework.6.1.3' from 'packages.config'
Successfully uninstalled 'EntityFramework.6.1.3' from HealthCare
Package 'EntityFramework.6.2.0' already exists in folder 'C:\Users\igor\Desktop\Folderi\C#\packages'
Added package 'EntityFramework.6.2.0' to 'packages.config'
Executing script file 'C:\Users\igor\Desktop\Folderi\C#\packages\EntityFramework.6.2.0\tools\init.ps1'
Executing script file 'C:\Users\igor\Desktop\Folderi\C#\packages\EntityFramework.6.2.0\tools\install.ps1'

Type 'get-help EntityFramework' to see all available Entity Framework commands.
Successfully installed 'EntityFramework 6.2.0' to HealthCare
Removing package 'EntityFramework.6.1.3' from folder 'C:\Users\igor\Desktop\Folderi\C#\packages'
Removed package 'EntityFramework.6.1.3' from folder 'C:\Users\igor\Desktop\Folderi\C#\packages'
PM>
```

Picture 2: Installing Entity Framework

Data Model

In the models folder we will create Pacijent entity class.

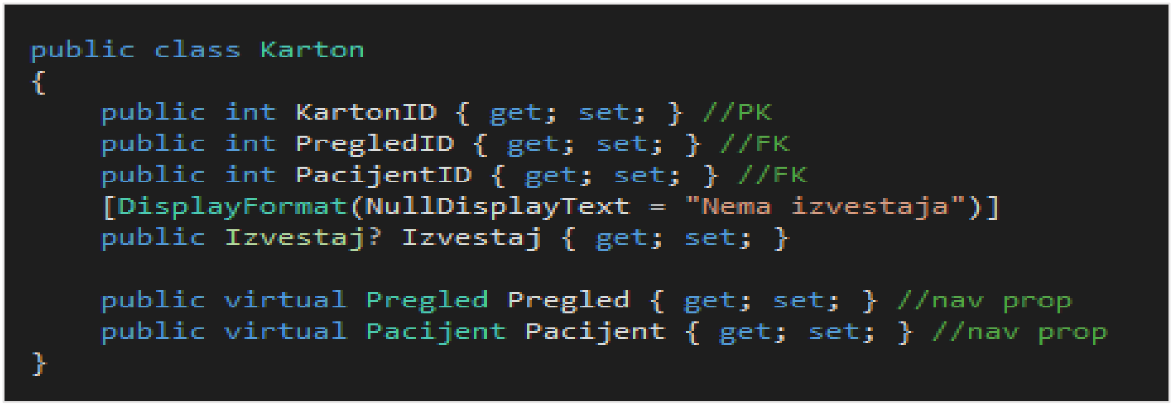


```
namespace HealthCare.Models
{
    public class Pacijent
    {
        public int ID { get; set; } //PK
        [Required]
        [StringLength(50)]
        [RegularExpression(@"^[a-zA-Z]+$", ErrorMessage = "Samo slova su dozvoljena.")]
        public string Prezime { get; set; }
        [Required]
        [StringLength(50)]
        [RegularExpression(@"^[a-zA-Z]+$", ErrorMessage = "Samo slova su dozvoljena.")]
        public string Ime { get; set; }
        [DataType(DataType.Date)]
        [DisplayFormat(DataFormatString = "{0:yyyy-MM-dd}", ApplyFormatInEditMode = true)]
        [Display(Name = "Datum Registracije")]
        public DateTime DatumRegistracije { get; set; }
        [Required]
        [Display(Name = "Naziv firme")]
        public string Firma { get; set; }

        [Display(Name = "Prezime i ime")]
        public string PunoIme { get { return Prezime + ", " + Ime; } }

        public virtual ICollection<Karton> Kartoni { get; set; }
        public virtual ICollection<File> Files { get; set; }
    }
```

Picture 3: Pacijent.cs

The ID property is a primary key column of the database table that corresponds to the Pacijent class. The Kartoni poperty is a navigation property, and it can contain one or more Karton entities. Its type is a list in which entries can be added, updated or deleted.

Next, in the models folder we will create Karton.cs class which contains two foreign keys PacijentID and LekarID and two navigation properites Pacijent and Lekar (can contain only one entity unlike Karton property in Pacijent class).

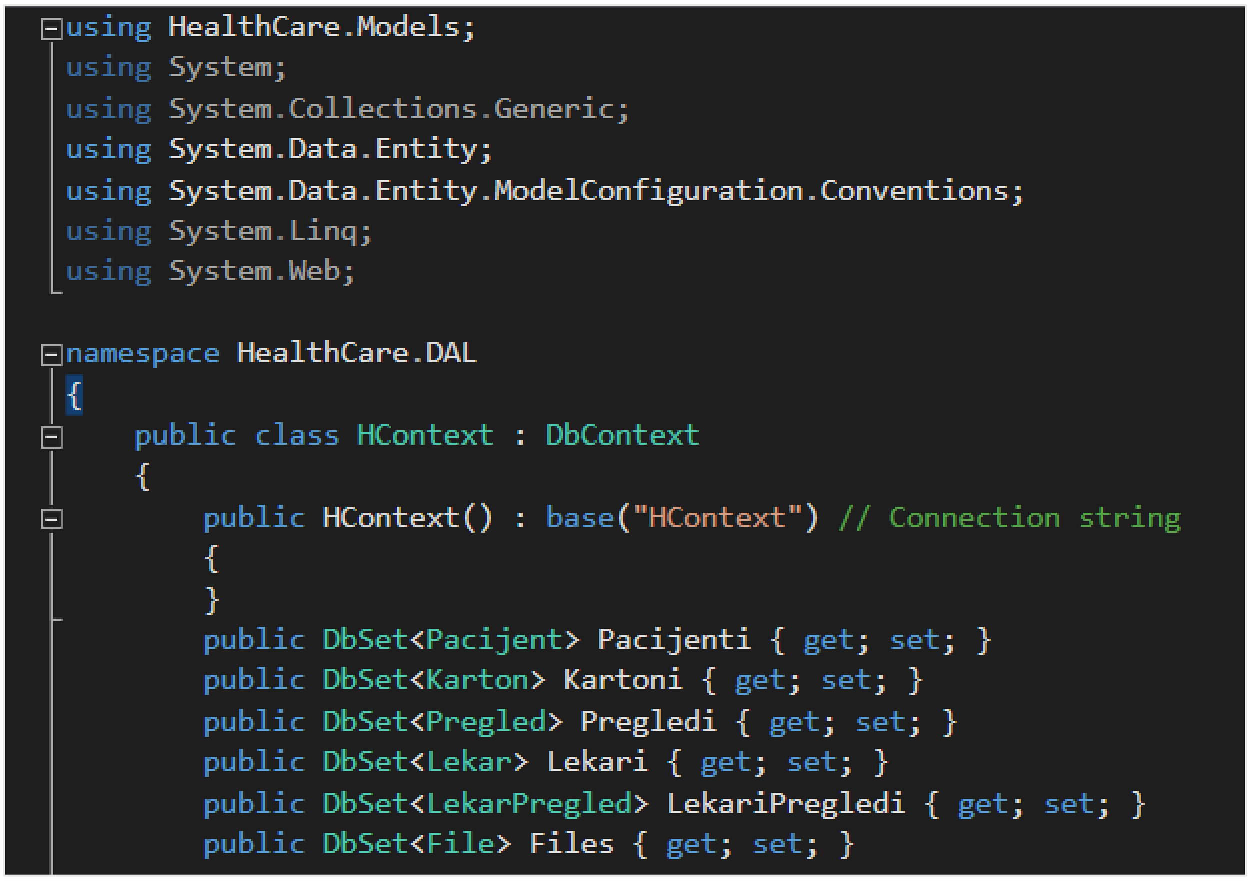


```
public class Karton
{
    public int KartonID { get; set; } //PK
    public int PregledID { get; set; } //FK
    public int PacijentID { get; set; } //FK
    [DisplayFormat(NullDisplayText = "Nema izvestaja")]
    public Izvestaj? Izvestaj { get; set; }

    public virtual Pregled Pregled { get; set; } //nav prop
    public virtual Pacijent Pacijent { get; set; } //nav prop
}
```

Picture 4: Karton.cs

Creating Hcontext class

Next, we will create Hcontext class, which was derived from the System.Data.Entity.DbContext class, that cordinates between the entity classes and the database.



```
using HealthCare.Models;
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Data.Entity.ModelConfiguration.Conventions;
using System.Linq;
using System.Web;

namespace HealthCare.DAL
{
    public class HContext : DbContext
    {
        public HContext() : base("HContext") // Connection string
        {
        }
        public DbSet<Pacijent> Pacijenti { get; set; }
        public DbSet<Karton> Kartoni { get; set; }
        public DbSet<Pregled> Pregledi { get; set; }
        public DbSet<Lekar> Lekari { get; set; }
        public DbSet<LekarPregled> LekariPregledi { get; set; }
        public DbSet<File> Files { get; set; }
```

Picture 5: HContext.cs

Connection string is passed in to the Hcontext class contructor:

```
public HContext() : base("HContext"){}
```

Each DbSet property in HContext class represents a collection of objects, which is mapped to a database table named same as the property.

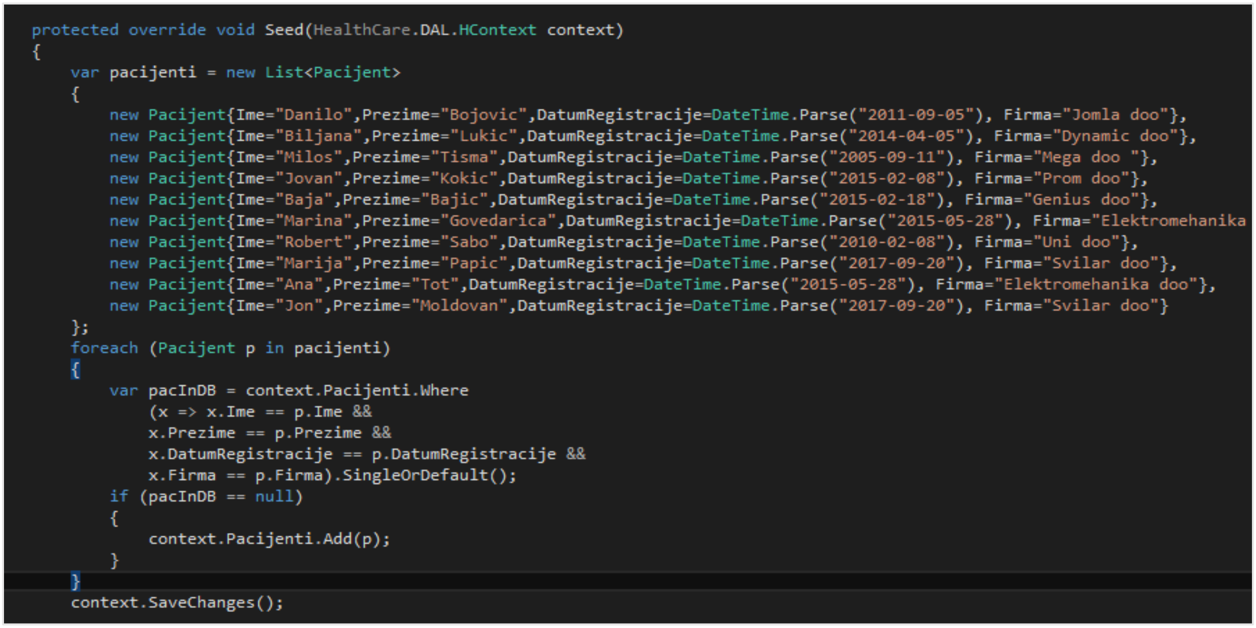
Code First Migrations

In the Package Manager Console we will add two commands:

```
enable-migrations
```

```
add-migration Mig1
```

Next, in the Configuration class we will add following code to the Seed method, which will load data to our database.



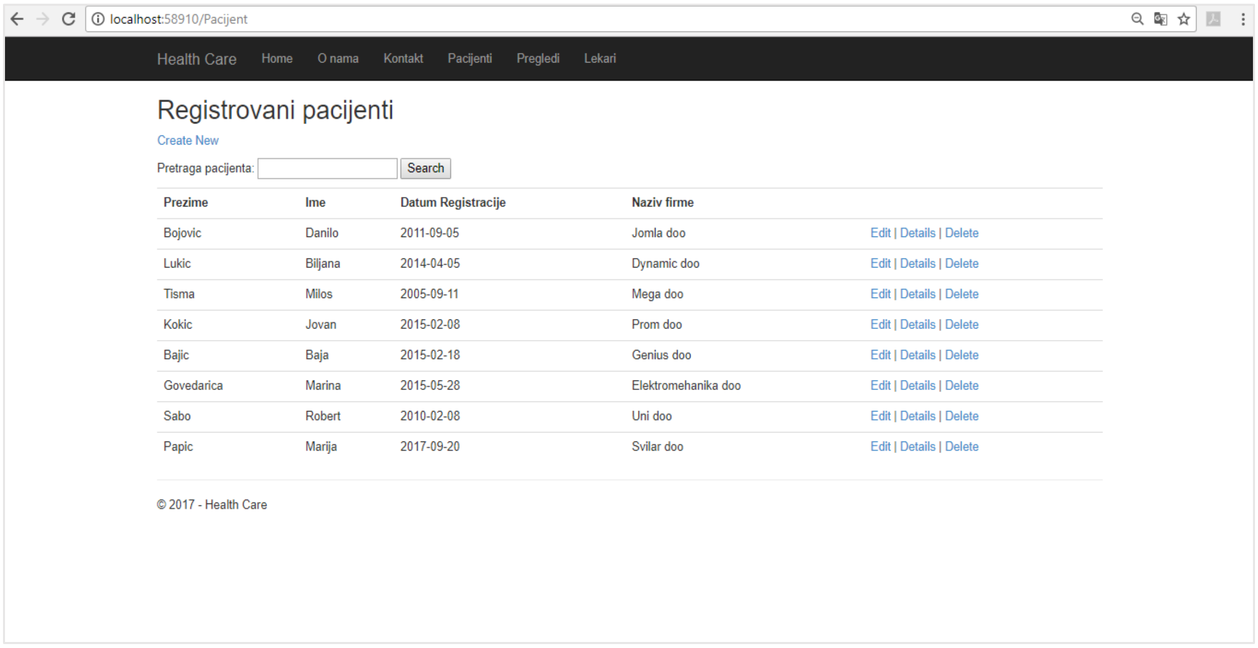
```
protected override void Seed(HealthCare.DAL.HContext context)
{
    var pacijenti = new List<Pacijent>
    {
        new Pacijent{Ime="Danilo",Prezime="Bojovic",DatumRegistracije=DateTime.Parse("2011-09-05"), Firma="Jomla doo"},
        new Pacijent{Ime="Biljana",Prezime="Lukic",DatumRegistracije=DateTime.Parse("2014-04-05"), Firma="Dynamic doo"},
        new Pacijent{Ime="Milos",Prezime="Tisma",DatumRegistracije=DateTime.Parse("2005-09-11"), Firma="Mega doo "},
        new Pacijent{Ime="Jovan",Prezime="Kokic",DatumRegistracije=DateTime.Parse("2015-02-08"), Firma="Prom doo"},
        new Pacijent{Ime="Baja",Prezime="Bajic",DatumRegistracije=DateTime.Parse("2015-02-18"), Firma="Genius doo"},
        new Pacijent{Ime="Marina",Prezime="Govedarica",DatumRegistracije=DateTime.Parse("2015-05-28"), Firma="Elektromehanika
        new Pacijent{Ime="Robert",Prezime="Sabo",DatumRegistracije=DateTime.Parse("2010-02-08"), Firma="Uni doo"},
        new Pacijent{Ime="Marija",Prezime="Papic",DatumRegistracije=DateTime.Parse("2017-09-20"), Firma="Svilar doo"},
        new Pacijent{Ime="Ana",Prezime="Tot",DatumRegistracije=DateTime.Parse("2015-05-28"), Firma="Elektromehanika doo"},
        new Pacijent{Ime="Jon",Prezime="Moldovan",DatumRegistracije=DateTime.Parse("2017-09-20"), Firma="Svilar doo"}
    };
    foreach (Pacijent p in pacijenti)
    {
        var pacInDB = context.Pacijenti.Where
            (x => x.Ime == p.Ime &&
            x.Prezime == p.Prezime &&
            x.DatumRegistracije == p.DatumRegistracije &&
            x.Firma == p.Firma).SingleOrDefault();
        if (pacInDB == null)
        {
            context.Pacijenti.Add(p);
        }
    }
    context.SaveChanges();
```

Picture 6: Configuration.cs

The Seed method takes database context object as an input parameter, and that context object is used to add new entities to the database.

After following the same procedure and creating collection of entities for each entity type and adding them to the corresponding DbSet property, we need to build the project and enter `update-database` command in the Package Manager Console window.

When we press CTRL + 5 to run the HealthCare project and click the Pacijent tab, we can see the data (that we previously added) the Seed method inserted.



Registrovani pacijenti

Create New

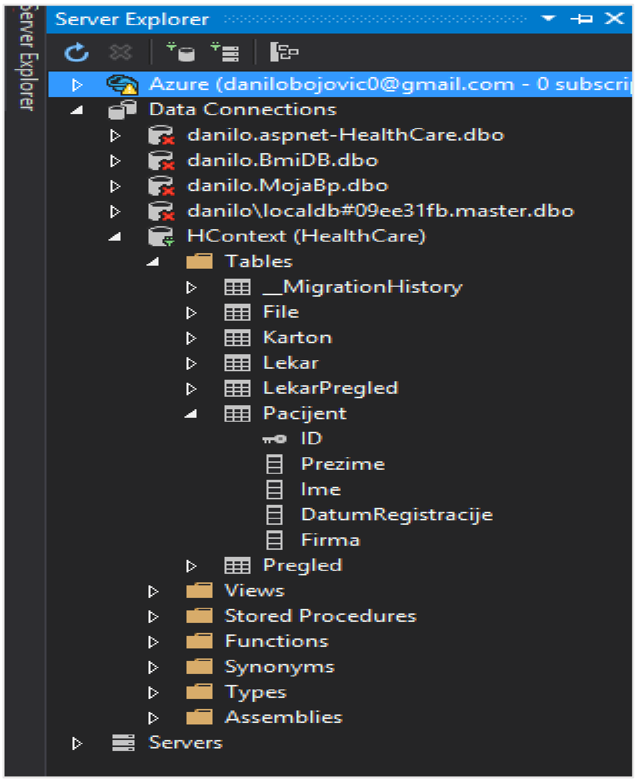
Pretraga pacijenta: Search

Prezime	Ime	Datum Registracije	Naziv firme	
Bojovic	Danilo	2011-09-05	Jomla doo	Edit | Details | Delete
Lukic	Biljana	2014-04-05	Dynamic doo	Edit | Details | Delete
Tisma	Milos	2005-09-11	Mega doo	Edit | Details | Delete
Kokic	Jovan	2015-02-08	Prom doo	Edit | Details | Delete
Bajic	Baja	2015-02-18	Genius doo	Edit | Details | Delete
Govedarica	Marina	2015-05-28	Elektromehanika doo	Edit | Details | Delete
Sabo	Robert	2010-02-08	Uni doo	Edit | Details | Delete
Papic	Marija	2017-09-20	Svilar doo	Edit | Details | Delete

© 2017 - Health Care

Picture 7: Pacijent/Index

In server explorer in Data Connections we can see Hcontext Database with tables.

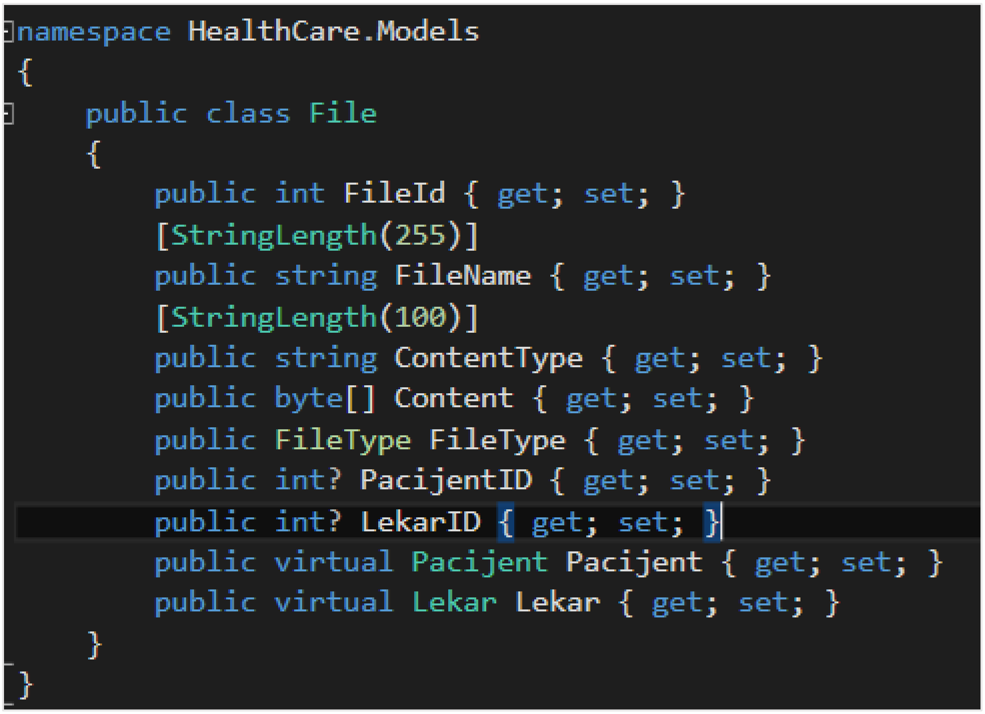


Picture 8: DataBase in Server Explorer

Adding files

In models folder File class was created to store files in our database tables.

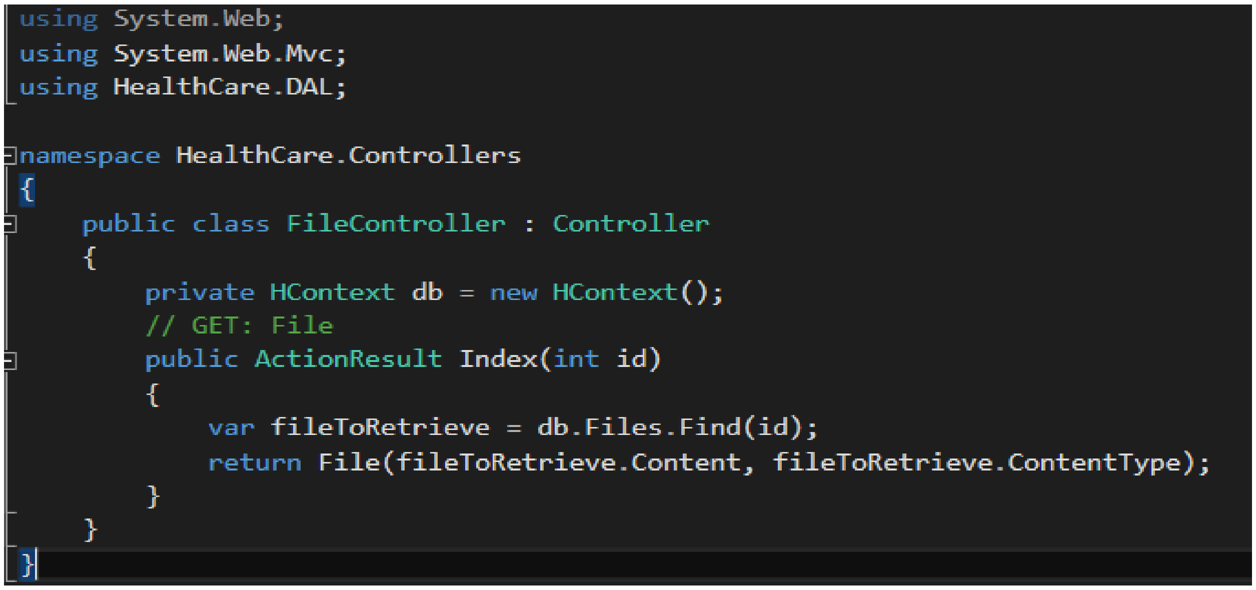
Class File has an virtual Pacijent property and virtual Lekar property, which is a way of establishing one to many relationship with both classes.



```
namespace HealthCare.Models
{
    public class File
    {
        public int FileId { get; set; }
        [StringLength(255)]
        public string FileName { get; set; }
        [StringLength(100)]
        public string ContentType { get; set; }
        public byte[] Content { get; set; }
        public FileType FileType { get; set; }
        public int? PacijentID { get; set; }
        public int? LekarID { get; set; }
        public virtual Pacijent Pacijent { get; set; }
        public virtual Lekar Lekar { get; set; }
    }
}
```

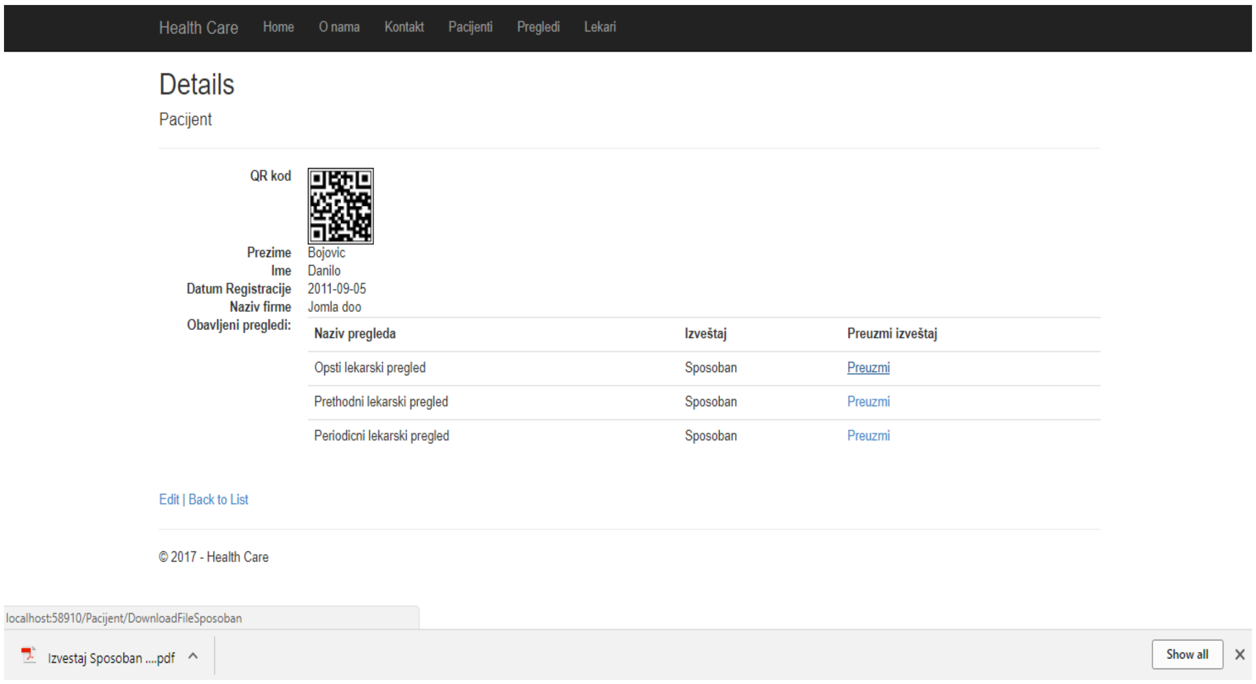
Picture 9: File.cs

After changing Create, Edit and Details methods and views we will add File Controller.



```
using System.Web;
using System.Web.Mvc;
using HealthCare.DAL;

namespace HealthCare.Controllers
{
    public class FileController : Controller
    {
        private HContext db = new HContext();
        // GET: File
        public ActionResult Index(int id)
        {
            var fileToRetrieve = db.Files.Find(id);
            return File(fileToRetrieve.Content, fileToRetrieve.ContentType);
        }
    }
}
```

Picture 10: FileController.cs


Health Care Home O nama Kontakt Pacijenti Pregledi Lekari

Details

Pacijent

QR kod	
Prezime	Bojovic
Ime	Danilo
Datum Registracije	2011-09-05
Naziv firme	Jomla doo
Obavljeni pregledi:	

Naziv pregleda	Izveštaj	Preuzmi izveštaj
Opsti lekarski pregled	Sposoban	Preuzmi
Prethodni lekarski pregled	Sposoban	Preuzmi
Periodicni lekarski pregled	Sposoban	Preuzmi

Edit | Back to List

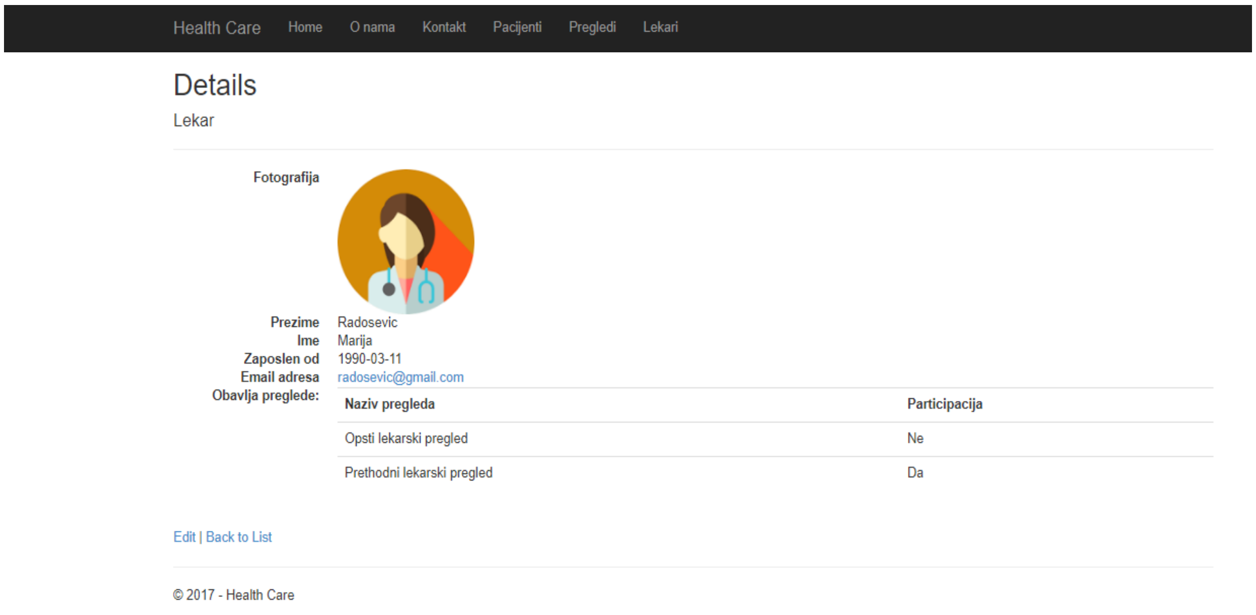
© 2017 - Health Care

localhost:58910/Pacijent/DownloadFileSposoban

Izvestaj Sposoban ....pdf

Show all

Picture 11: Pacijent details page


Health Care Home O nama Kontakt Pacijenti Pregledi Lekari

Details

Lekar

Fotografija	
Prezime	Radosevic
Ime	Marija
Zaposlen od	1990-03-11
Email adresa	radosevic@gmail.com
Obavlja preglede:	

Naziv pregleda	Participacija
Opsti lekarski pregled	Ne
Prethodni lekarski pregled	Da

Edit | Back to List

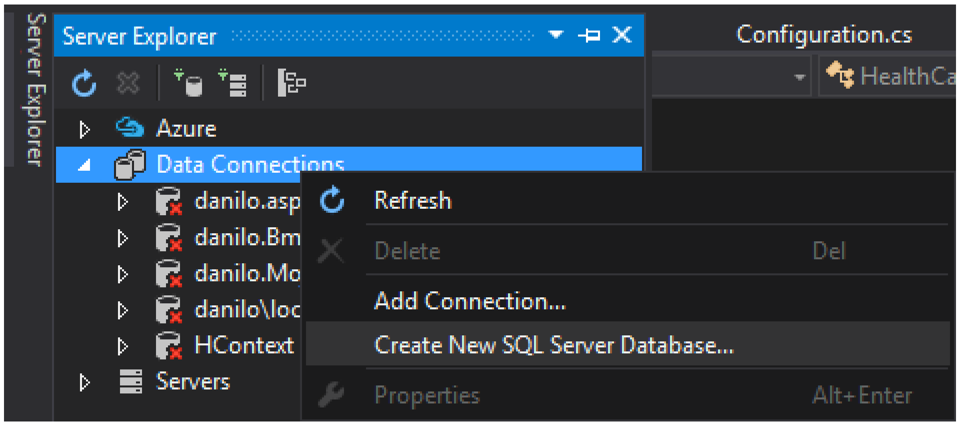
© 2017 - Health Care

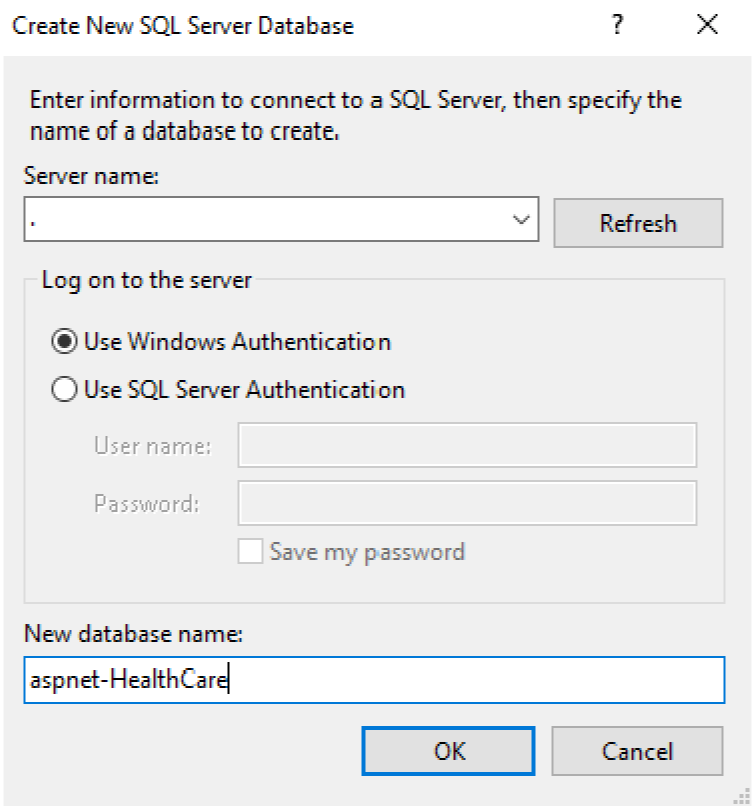
Picture 12: Lekari details page

Deploying application

Next, we will deploy HealthCare web application to Internet Information Services on the local computer.

After installing IIS and changing .NET Framework version to .NET v4.5, in the Server Explorer menu click to Data Connections and create new SQL Server database. As shown in picture below, in new SQL Server database dialog box we will choose Windows Authentication and specify our Server name (dot represents our local server) and new database name.




Create New SQL Server Database

Enter information to connect to a SQL Server, then specify the name of a database to create.

Server name: .

Refresh

Log on to the server

[✓ Use Windows Authentication]

Use SQL Server Authentication

User name:

Password:

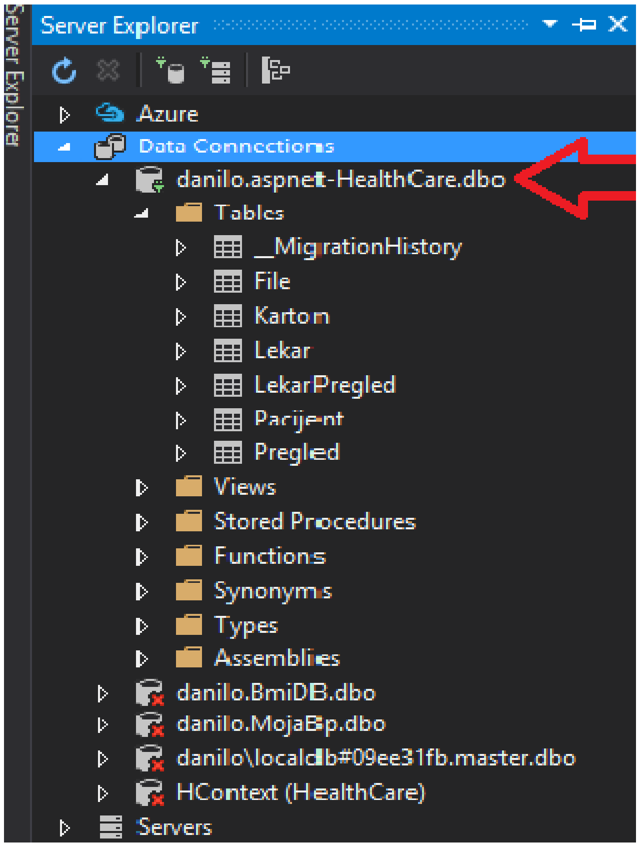
Save my password

New database name: aspnet-HealthCare

OK Cancel

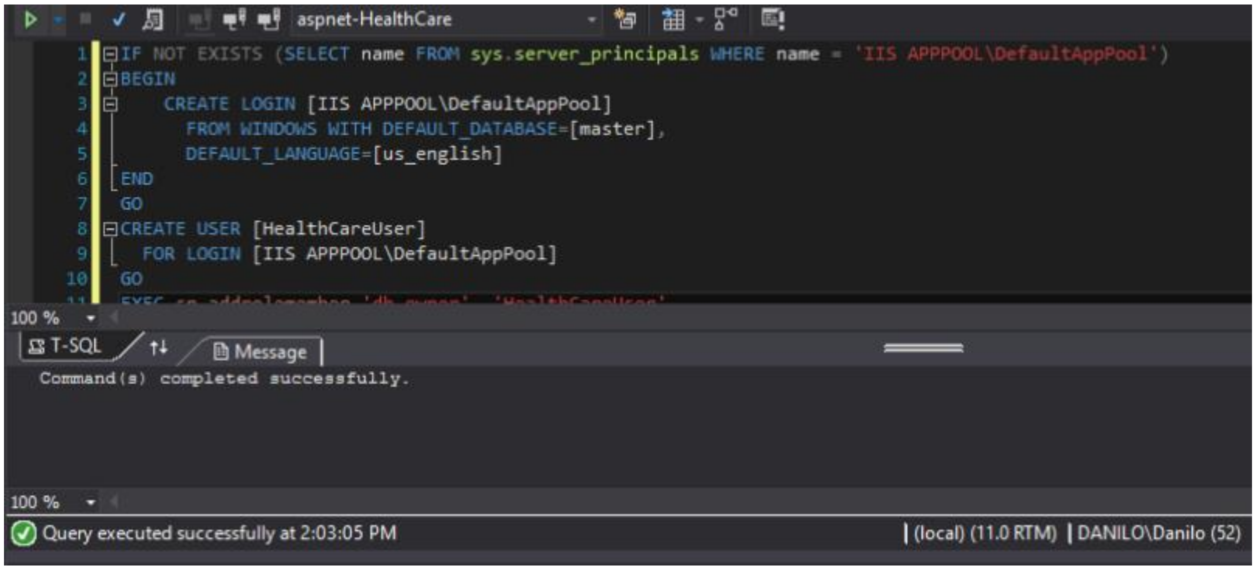
Picture 13, 14: Create New SQL Server Database

Server Expoler shows our new Database.



Picture 15: New Database

Next, we will create and run grant script for the new database.



```
IF NOT EXISTS (SELECT name FROM sys.server_principals WHERE name = 'IIS APPPOOL\DefaultAppPool')
BEGIN
    CREATE LOGIN [IIS APPPOOL\DefaultAppPool]
      FROM WINDOWS WITH DEFAULT_DATABASE=[master],
      DEFAULT_LANGUAGE=[us_english]
END
GO
CREATE USER [HealthCareUser]
  FOR LOGIN [IIS APPPOOL\DefaultAppPool]
GO
EXEC sp_addrolemember 'db_owner', 'HealthCareUser'
```

```
Command(s) completed successfully.
```

Query executed successfully at 2:03:05 PM | (local) (11.0 RTM) | DANILO\Danilo (52)

Picture 16: Grant script

Next step is to create the publish profile. In Solution explorer right click on HealthCare project and select publish. In the publish web dialog box select new Custom and after entering information about connection click Validate Connection.

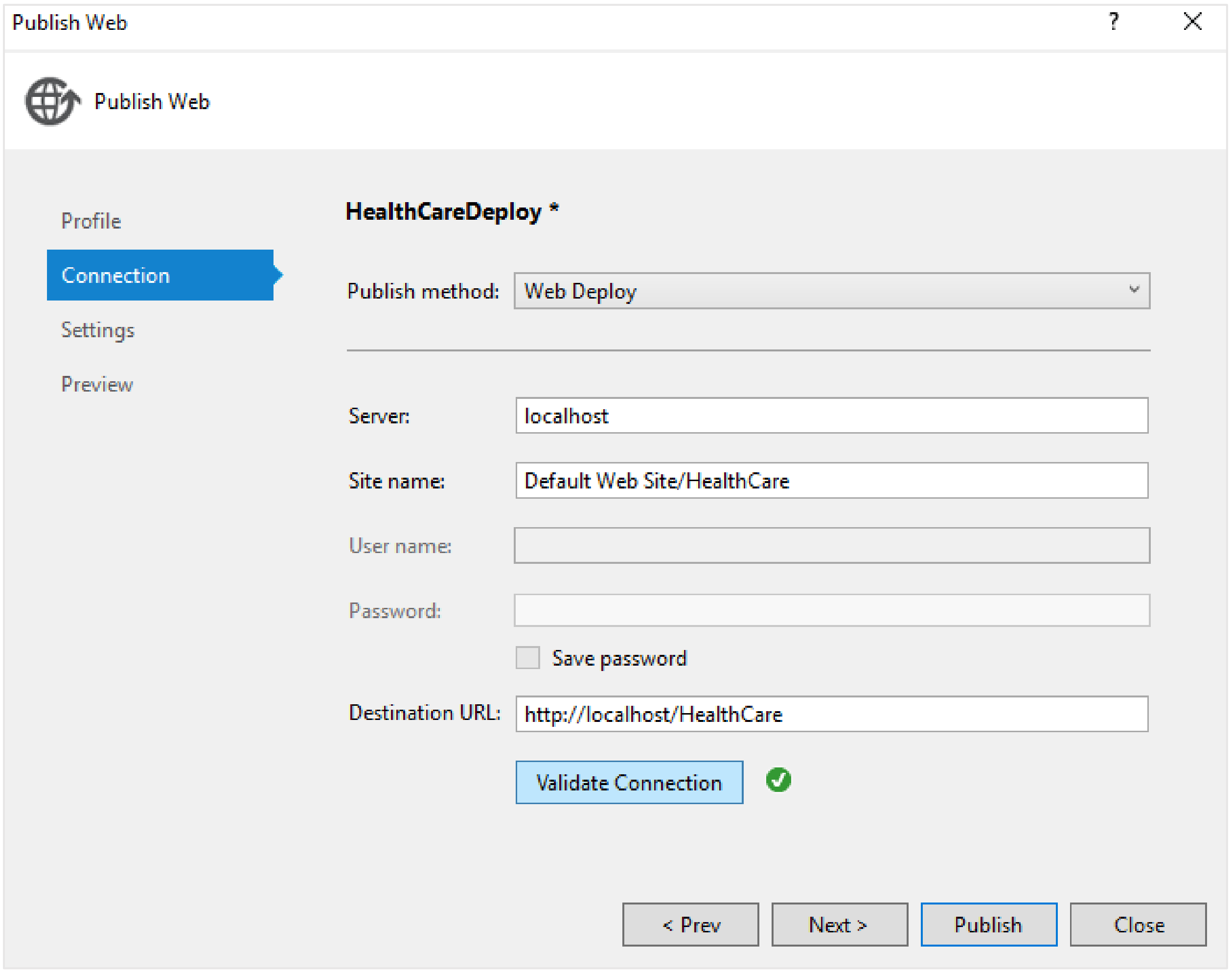
In the Setting menu we will expand File Publish Options, and then select Exclude files from the App_Data folder. In Hcontext box we will add connection string that points to our aspnet-HealthCare database:

```
Data Source=.;Initial Catalog=aspnet-HealthCare;Integrated Security=True;Pooling=False
```

After adding config transform, we will insert following code to Web.HealthCareDeploy.config transform file:

```
<appSettings>
       <add  key="Environment"  value="HealthCareDeploy"  xdt:Transform="SetAttributes"
       xdt:Locator="Match(key)"/>
</appSettings>
```

Last step is to click on Start Preview on Preview tab, to see files that will be copied, and Publish our site on http://localhost/HealthCare.



HealthCareDeploy *

Publish method: Web Deploy

Server: localhost

Site name: Default Web Site/HealthCare

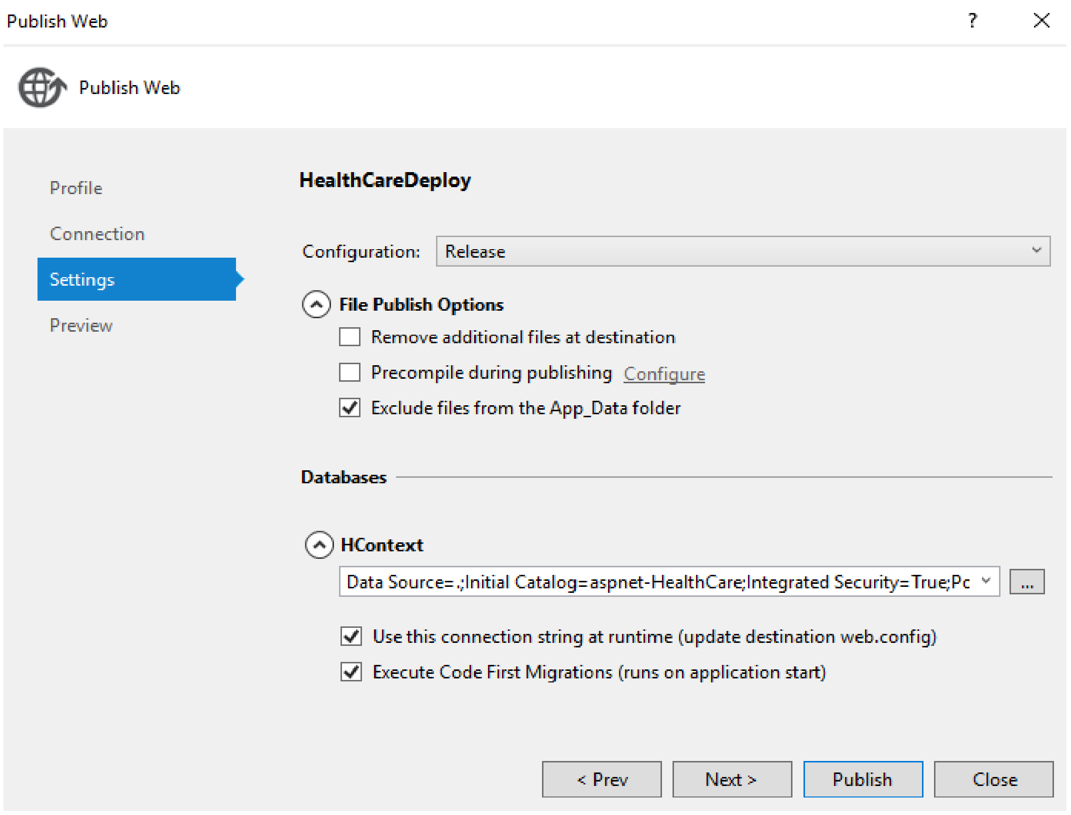
User name:

Password:

Save password

Destination URL: http://localhost/HealthCare

Validate Connection ✓



HealthCareDeploy

Configuration: Release

File Publish Options

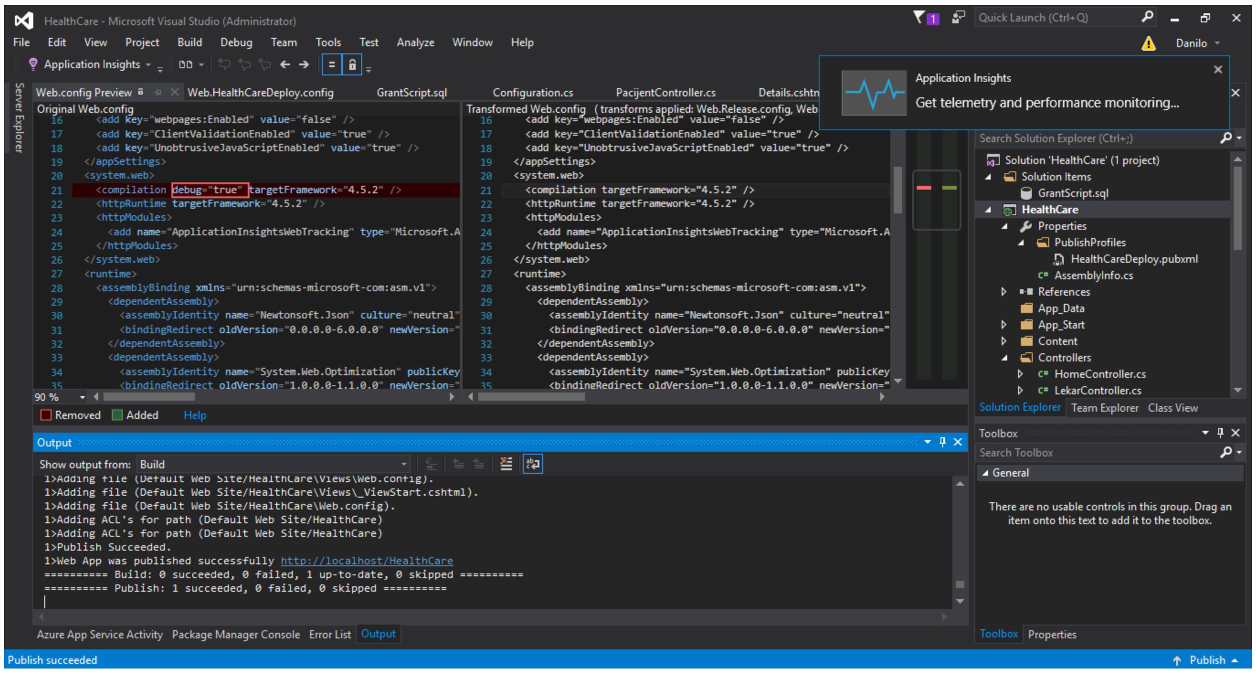
- [ ] Remove additional files at destination
- [ ] Precompile during publishing Configure
- [x] Exclude files from the App_Data folder

Databases

HContext

Data Source=.;Initial Catalog=aspnet-HealthCare;Integrated Security=True;Pc

- [x] Use this connection string at runtime (update destination web.config)
- [x] Execute Code First Migrations (runs on application start)



Original Web.config

```
    <add key="webpages:Enabled" value="false" />
    <add key="ClientValidationEnabled" value="true" />
    <add key="UnobtrusiveJavaScriptEnabled" value="true" />
  </appSettings>
  <system.web>
    <compilation debug="true" targetFramework="4.5.2" />
    <httpRuntime targetFramework="4.5.2" />
    <httpModules>
      <add name="ApplicationInsightsWebTracking" type="Microsoft.A
    </httpModules>
  </system.web>
  <runtime>
    <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
      <dependentAssembly>
        <assemblyIdentity name="Newtonsoft.Json" culture="neutral"
        <bindingRedirect oldVersion="0.0.0.0-6.0.0.0" newVersion="
      </dependentAssembly>
      <dependentAssembly>
        <assemblyIdentity name="System.Web.Optimization" publicKey
        <bindingRedirect oldVersion="1.0.0.0-1.1.0.0" newVersion="
```

Transformed Web.config (transforms applied: Web.Release.config, Web

```
    <add key="webpages:Enabled" value="false" />
    <add key="ClientValidationEnabled" value="true" />
    <add key="UnobtrusiveJavaScriptEnabled" value="true" />
  </appSettings>
  <system.web>
    <compilation targetFramework="4.5.2" />
    <httpRuntime targetFramework="4.5.2" />
    <httpModules>
      <add name="ApplicationInsightsWebTracking" type="Microsoft.A
    </httpModules>
  </system.web>
  <runtime>
    <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
      <dependentAssembly>
        <assemblyIdentity name="Newtonsoft.Json" culture="neutral"
        <bindingRedirect oldVersion="0.0.0.0-6.0.0.0" newVersion="
      </dependentAssembly>
      <dependentAssembly>
        <assemblyIdentity name="System.Web.Optimization" publicKey
        <bindingRedirect oldVersion="1.0.0.0-1.1.0.0" newVersion="
```

Output

Show output from: Build

```
1>Adding file (Default Web Site/HealthCare\Views\Web.config).
1>Adding file (Default Web Site/HealthCare\Views\_ViewStart.cshtml).
1>Adding file (Default Web Site/HealthCare\Web.config).
1>Adding ACL's for path (Default Web Site/HealthCare)
1>Adding ACL's for path (Default Web Site/HealthCare)
1>Publish Succeeded.
1>Web App was published successfully http://localhost/HealthCare
========== Build: 0 succeeded, 0 failed, 1 up-to-date, 0 skipped ==========
========== Publish: 1 succeeded, 0 failed, 0 skipped ==========
```

Picture 17, 18, 19: Publish to IIS