1. Exam.cs-model

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace ExamDI.Models

{

public class Exam

{

public int ID { get; set; }

public string Name { get; set; }

public DateTime Data { get; set; }

public int NumberRoom { get; set; }

}

}

1. ApplicationDbContext.cs

using ExamDI.Models;

using Microsoft.AspNetCore.Identity.EntityFrameworkCore;

using Microsoft.EntityFrameworkCore;

using System;

using System.Collections.Generic;

using System.Text;

namespace ExamDI.Data

{

public class ApplicationDbContext : IdentityDbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)

: base(options)

{

}

public DbSet<Exam> Exams { get; set; }

}

}

1. Миграцията

Add-Migration InitialCreate

Update database

1. Добавя се контролер с MVC
2. Добавя се скеле с Identity
3. Сървиси

public void ConfigureServices(IServiceCollection services)

{

services.AddIdentity<IdentityUser, IdentityRole>()

.AddEntityFrameworkStores<ApplicationDbContext>()

.AddDefaultTokenProviders()

.AddDefaultUI();

services.AddMvc().SetCompatibilityVersion(CompatibilityVersion.Version\_3\_0);

services.Configure<IdentityOptions>(options =>

{

// Password settings.

options.Password.RequireDigit = false;

options.Password.RequireLowercase = false;

options.Password.RequireNonAlphanumeric = false;

options.Password.RequireUppercase = false;

options.Password.RequiredLength = 3;

options.Password.RequiredUniqueChars = 0;

options.User.RequireUniqueEmail = true;

});

}

................

public void Configure(IApplicationBuilder app, IWebHostEnvironment env)

{.....................

using (var serviceScope = app.ApplicationServices.CreateScope())

{

using (var context = serviceScope.ServiceProvider.GetRequiredService<ApplicationDbContext>())

{

context.Database.Migrate();

if (!context.Roles.Any())

{

context.Roles.Add(new IdentityRole

{

Name = "Admin",

NormalizedName = "ADMIN",

});

context.Roles.Add(new IdentityRole

{

Name = "User",

NormalizedName = "USER",

});

context.SaveChanges();

}

}

}

1. В Register.cs

public async Task<IActionResult> OnPostAsync(string returnUrl = null)

{……………

if (ModelState.IsValid)

{

var user = new IdentityUser { UserName = Input.UserName, Email = Input.Email };

var result = await \_userManager.CreateAsync(user, Input.Password);

if (result.Succeeded)

{

\_logger.LogInformation("User created a new account with password.");

if (\_userManager.Users.Count() == 1)

{ await \_userManager.AddToRoleAsync(user, "Admin"); }

else

{ await \_userManager.AddToRoleAsync(user, "User"); }

1. Авторизация Index.cshtml

@if(!this.User.Identity.IsAuthenticated)

{

<div class="text-center">

<h1 class="display-4">Welcome! Pleace Identity</h1>

<p>Learn about <a href="https://docs.microsoft.com/aspnet/core">building Web apps with ASP.NET Core</a>.</p>

</div>

}

else

{

if (this.User.IsInRole("Admin"))

{<div><h2>Zdravej ADMIN</h2></div>

}

else {

<div class="jumbotron bg-eventures w-75 mx-auto">

<h1 class="text-center text-secondary">Welcome, @this.User.Identity.Name</h1>

</div>

}

}

1. Авторизация \_Layout.cshtml

@if (this.User.Identity.IsAuthenticated)

{

<li class="nav-item active">

<div class="dropdown show">

<a class="nav-link active dropdown-toggle" href="#"

id="dropdownMenuLink"

data-toggle="dropdown"

aria-haspopup="true"

aria-expanded="false">

Exam

</a>

<div class="dropdown-menu" aria-labelledby="dropdownMenuLink">

<a class="dropdown-item" href="/Exams/All">All Exams</a>

@if (this.User.IsInRole("Admin"))

{

<a class="dropdown-item" href="/Exams">Edit Exams</a>

<a class="dropdown-item" href="/Exams/Create">Create Exam</a>}

</div>

</div>

</li>}

</ul>

<ul class="navbar-nav">

@if (!this.User.Identity.IsAuthenticated)

{

<li class="nav-item active">

@\*<a class="nav-link" href="/Identity/Account/Login">Login</a>\*@

</li>

<li class="nav-item active">

@\*<a class="nav-link" href="/Identity/Account/Register">Register</a>\*@

</li> }

else

{

<li class="nav-item active">

@\*<a class="nav-link" href="/Identity/Account/Logout">Logout</a>\*@

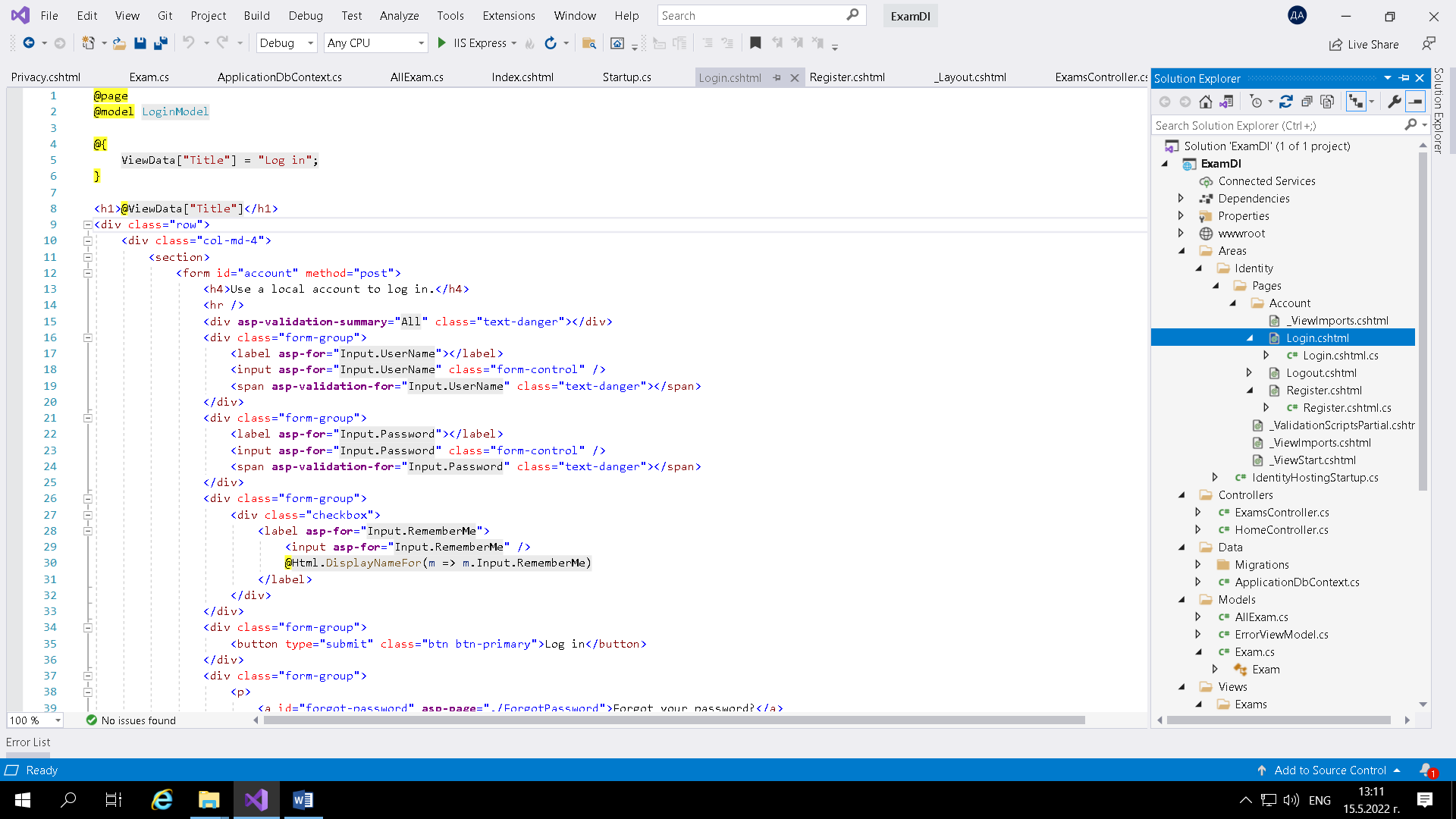
</li>}

</ul>

</div>

</nav>

1. Автентикацията в Login с username I password



public class InputModel

{

[Required]

public string UserName { get; set; }

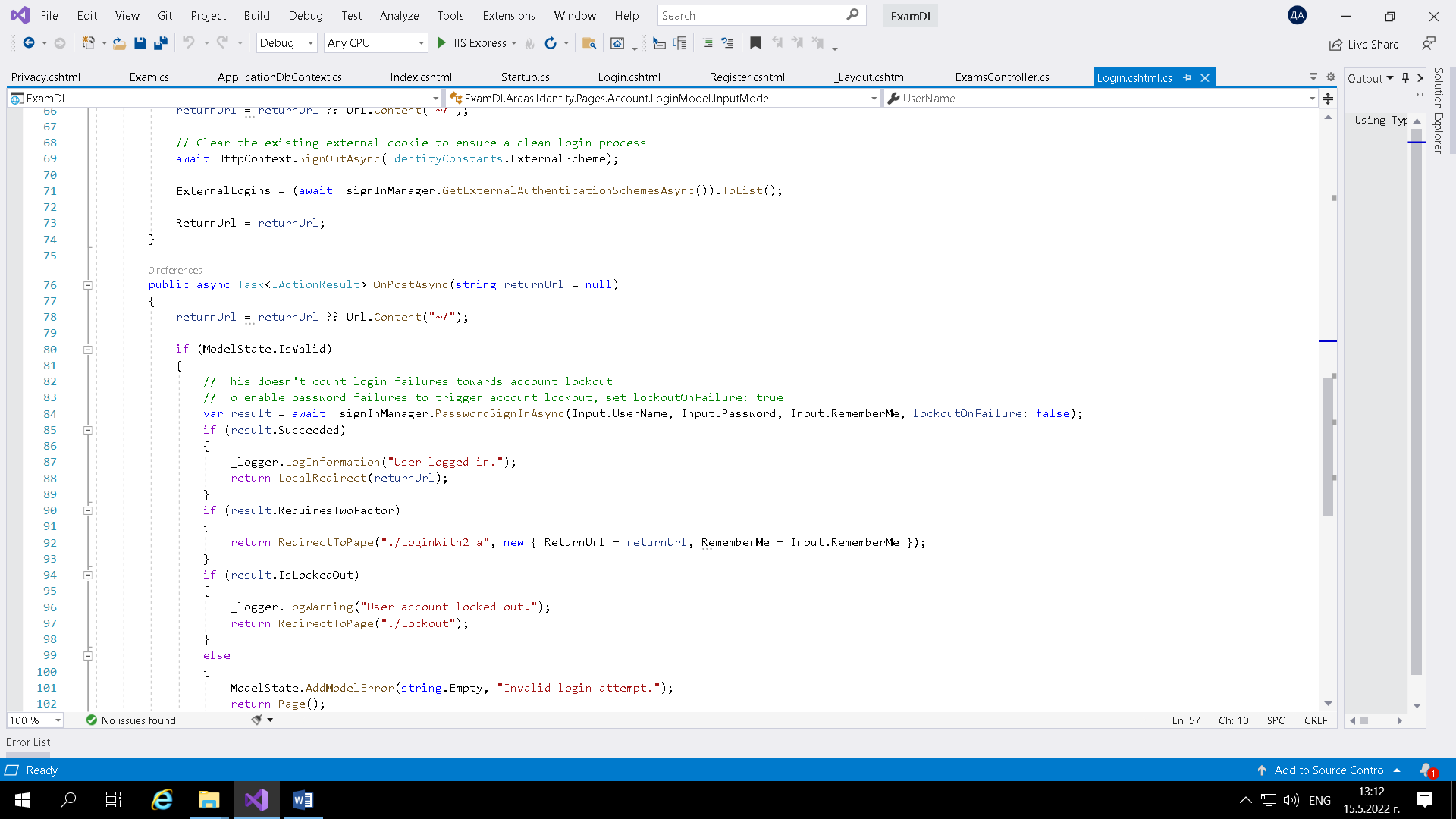
[Required]

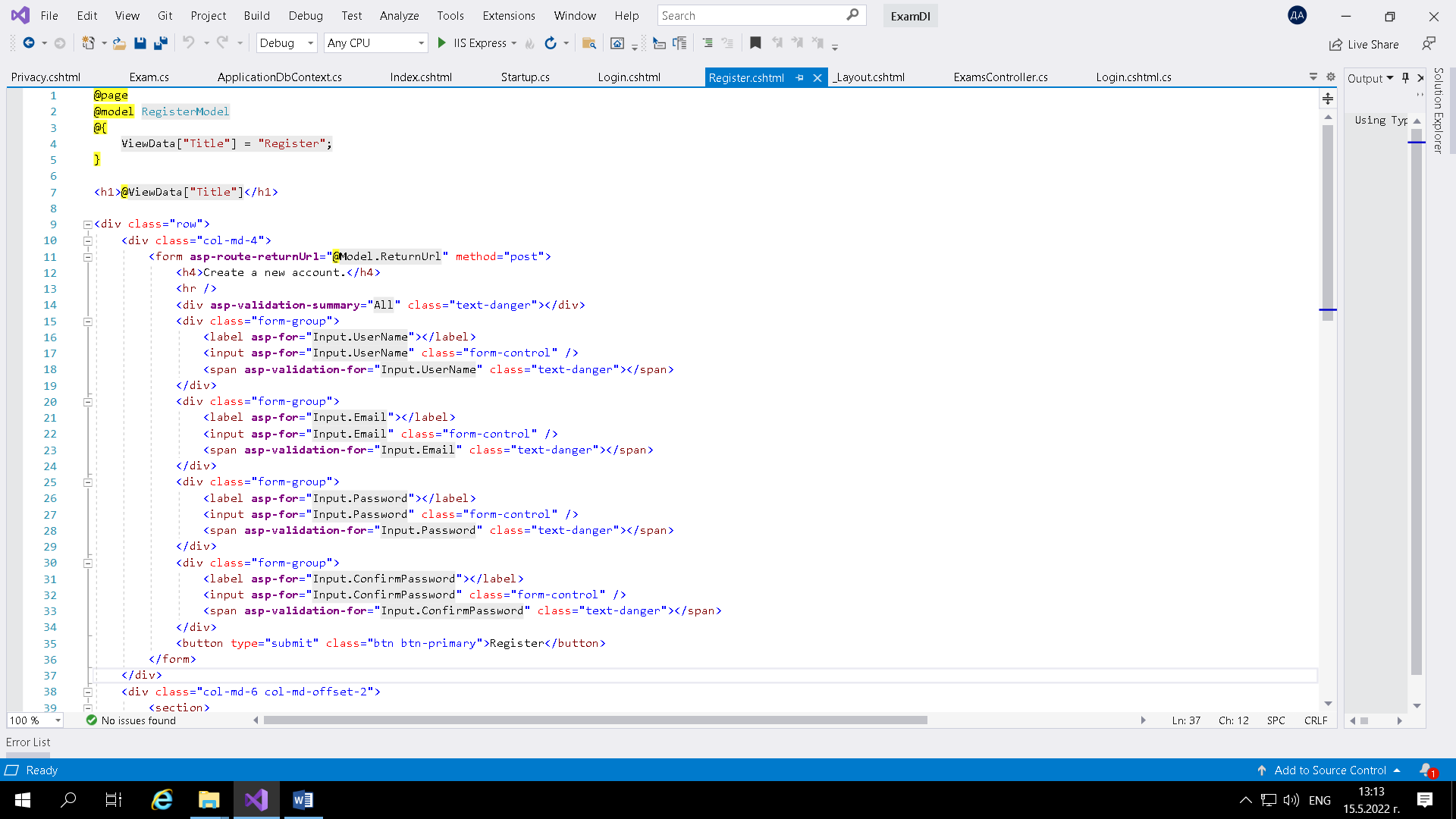
[DataType(DataType.Password)]

public string Password { get; set; }

[Display(Name = "Remember me?")]

public bool RememberMe { get; set; }

}

1. Автентикация Регистър

public class InputModel

{

[Required]

[Display(Name = "UserName")]

public string UserName { get; set; }

[Required]

[EmailAddress]

[Display(Name = "Email")]

public string Email { get; set; }

[Required]

[StringLength(100, ErrorMessage = "The {0} must be at least {2} and at max {1} characters long.", MinimumLength = 6)]

[DataType(DataType.Password)]

[Display(Name = "Password")]

public string Password { get; set; }

[DataType(DataType.Password)]

[Display(Name = "Confirm password")]

[Compare("Password", ErrorMessage = "The password and confirmation password do not match.")]

public string ConfirmPassword { get; set; }

}

