1. Exam.cs-model

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace ExamDI.Models

{

Може да има например за [Range(100,110)] или [StringLength(20)].

Създава се допълнителен клас който се казва нали както е зададено в задачата за типове на изброяване например TypeRoom, там променяме класа и го кръщаваме горе public enum TypeRoom, и изброяваме това което ни е зададено. Дефинира в нормалния клас public TypeRoom TypeRoom {get; set;}

Пример за един нормален модел: Името е според задачата на класа както и public enum-a после.

public class Room

{

public int Id { get; set; }

[Range(100,110)]

public int NumberOfRoom { get; set; }

public decimal Price { get; set; }

public string Image { get; set; }

public TypeRoom TypeRoom { get; set; }

public int NumberOfNights {get; set;}

public DateTime Start { get; set; }

public DateTime End { get; set; }

}

}

public enum TypeRoom

{

Edinichna, dvoichna , studio

}

1. Controller
2. Миграцията

Add-Migration InitialCreate

Update-DataBase

4.Layout

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

{

<li class="nav-item">

<a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Index">Home</a>

</li>

<li class="nav-item">

<a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Privacy">Privacy</a>

</li>

}

else

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

{

<li class="nav-item">

<a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Index">Home</a>

</li>

<li class="nav-item">

<a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Privacy">Privacy</a>

</li>

<li class="nav-item">

<a class="nav-link text-dark" asp-area="" asp-controller="Rooms" asp-action="Index">DanniZaStaq</a>

</li>

<li class="nav-item">

<a class="nav-link text-dark" asp-area="" asp-controller="Rooms" asp-action="IndexKlient">StaiZaKlient</a>

</li>

}

else

{

<li class="nav-item">

<a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Index">Home</a>

</li>

<li class="nav-item">

<a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Privacy">Privacy</a>

</li>

<li class="nav-item">

<a class="nav-link text-dark" asp-area="" asp-controller="Rooms" asp-action="IndexKlient">StaiZaKlient</a>

</li>

}

</ul>

1. Добавя се скеле с Identity - Отиваме Areas-> Add -> New Scaffolded item -> Identity -> Add -> след това избираме Log in, Log out, Register, отдолу DbContext

register.cshtml.cs

поставяме public string UserName { get; set; } След това слизаме надолу и търсим Input UserName = UserName

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"); }

Login.cs и останалите

placeholder =" UserName">

1. Сървиси

public Startup(IConfiguration configuration)

{

Configuration = configuration;

}

public IConfiguration Configuration { get; }

// This method gets called by the runtime. Use this method to add services to the container.

public void ConfigureServices(IServiceCollection services)

{

services.AddDbContext<ApplicationDbContext>(options =>

options.UseSqlServer(

Configuration.GetConnectionString("DefaultConnection")));

//services.AddDefaultIdentity<IdentityUser>(options => options.SignIn.RequireConfirmedAccount = true)

// .AddEntityFrameworkStores<ApplicationDbContext>();

{

services.AddIdentity<IdentityUser, IdentityRole>().AddEntityFrameworkStores<ApplicationDbContext>().AddDefaultTokenProviders().AddDefaultUI();

services.Configure<IdentityOptions>(options =>

{

options.Password.RequireDigit = false;

options.Password.RequireLowercase = false;

options.Password.RequireNonAlphanumeric = false;

options.Password.RequireUppercase = false;

options.Password.RequiredLength =

options.Password.RequiredUniqueChars = 0;

options.User.RequireUniqueEmail = true;

});

}

services.AddControllersWithViews();

services.AddRazorPages();

}

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();

}

if (env.IsDevelopment())

{

app.UseDeveloperExceptionPage();

app.UseDatabaseErrorPage();

}

else

{

app.UseExceptionHandler("/Home/Error");

app.UseHsts();

}

app.UseHttpsRedirection();

app.UseStaticFiles();

app.UseRouting();

app.UseAuthentication();

app.UseAuthorization();

app.UseEndpoints(endpoints =>

{

endpoints.MapControllerRoute(

name: "default",

pattern: "{controller=Home}/{action=Index}/{id?}");

endpoints.MapRazorPages();

});

}

}

}

}

}

1. Авторизация Index.cshtml

<style>

img {

width: 50%;

border: 2px solid blue;

}

.jumbotron{background-color:lightgreen}

#sexy {

width: 50%;

border: 2px solid pink;

margin-left: 256px;

}

#sexyy {

width: 50%;

border: 2px solid blue;

margin-left: 200px;

}

.display-4

{

text-align:center;

color: pink;

}

h2

{

color:lightpink;

text-align:center;

}

</style>

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

{

<div class="text-center">

<h1 class="display-4">Добре дошли! Моля идентифицирайте се!</h1>

<img src="~/Images/sexy.jpg" />

</div>

}

else

{

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

{<div><h2>Have a nice day, Admin!</h2></div>

<img id="sexy" src="~/Images/ulala.jpg" />

}

else

{

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

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

<h1 class="display-4">Тук се пише текст специално за задачата: </h1>

<p>

<h4>Тук се пише текст специално за задачата: </h4>

Тук се пише текст специално за задачата:

</p>

<img id ="sexyy"src="~/Images/sexy.jpg" />

</div>

}

}

**Create единствено добавяш опциите си от класа за енума:**

<select asp-for="TypeRoom" class="form-control">

<option>Edinichna</option>

<option>Dvoina</option>

<option>Studio</option>

</select>

**При Details там** трябва да имаш два един за админа другия за клиента кръщаваш втория DetailsKlient:

В DetailsKlient си добавяш

<style>

img {

width: 50%;

}

</style>

</style>

**После слизаш надолу и буташ втория ModelImage**:

@\*@Html.DisplayFor(model => model.Image)\*@ слагаш само него в коментар единия ред и отдолу пишеш

<img src="@Model.Image" />

Слизаш най надолу и махаш онези бутончета Edit Delete и т.н оставяш само

<a asp-action="IndexKlient">Back to List</a> обърни внимание на името на индекса трябва да ти е на клиента.

**След това отиваш в другия истинския Details за админа**. И там по същия начин всичко както в клиента

**В Edit та**м единствено добавяш опциите си от класа за енума

**Главен индекс**

<th>Трайност до края на промоцията</th>

След това минаваме на долните пак отново след foreach-a и отиваме на Image и там реда го отбелязваме @\*@Html.DisplayFor(modelItem => item.Image)\*@ и го слагаме в коментар: и отдолу пишем: <img src="@item.Image" />

След това най - долу създаваме ново <td>:

<td>

@(item.End.Day - item.Start.Day ) days

</td>

И най най отдолу ако искам да имам бутончета поставям това :

<td>

<button id="buton"> <a asp-action="Edit" asp-route-id="@item.Id">Edit</a> </button>

</td>

<td><button id="buton"> <a asp-action="Details" asp-route-id="@item.Id">Details</a></button></td>

<td><button id="buton"> <a asp-action="Delete" asp-route-id="@item.Id">Delete</a></button></td>

}

</tbody>

</table>

Преминаваме към IndexKlient: В него в началото пак в <thead> добавяме <th>Трайност до края на промоцията</th> и слагаме в коментар za datata start и end

В foreach по същия начин : @\*@Html.DisplayFor(modelItem => item.Image)\*@ и го слагаме в коментар: и отдолу пишем: <img src="@item.Image" /> , там пак датата и края на датата се слагат в коментар, и се поставя същото <td> от предишния индекс:

<td>

@(item.End.Day - item.Start.Day ) days

</td>

Датите се слагат в коментар!

Най - отдолу

<td>

@\*<a asp-action="Edit" asp-route-id="@item.Id">Edit</a> |\*@

<a asp-action="DetailKlient" asp-route-id="@item.Id">Details</a> |

@\*<a asp-action="Delete" asp-route-id="@item.Id">Delete</a>\*@

</td>

**Относно стилизацията: Примерна за Layout:**

<style>

nav {

background-image: url(/Images/sexy.jpg);

}

body {

background-image: url(/Images/sexy.jpg);

}

</style>

**!. това е ако не успея и не ми се занимава**:

<style>

.container {

background-color: mistyrose;

color: aquamarine;

background-image:url(/Images/sexy.jpg);

background-size: cover;

}

div {

background-color: cornflowerblue;

}

footer {

background-color: darkgrey;

color: red;

}

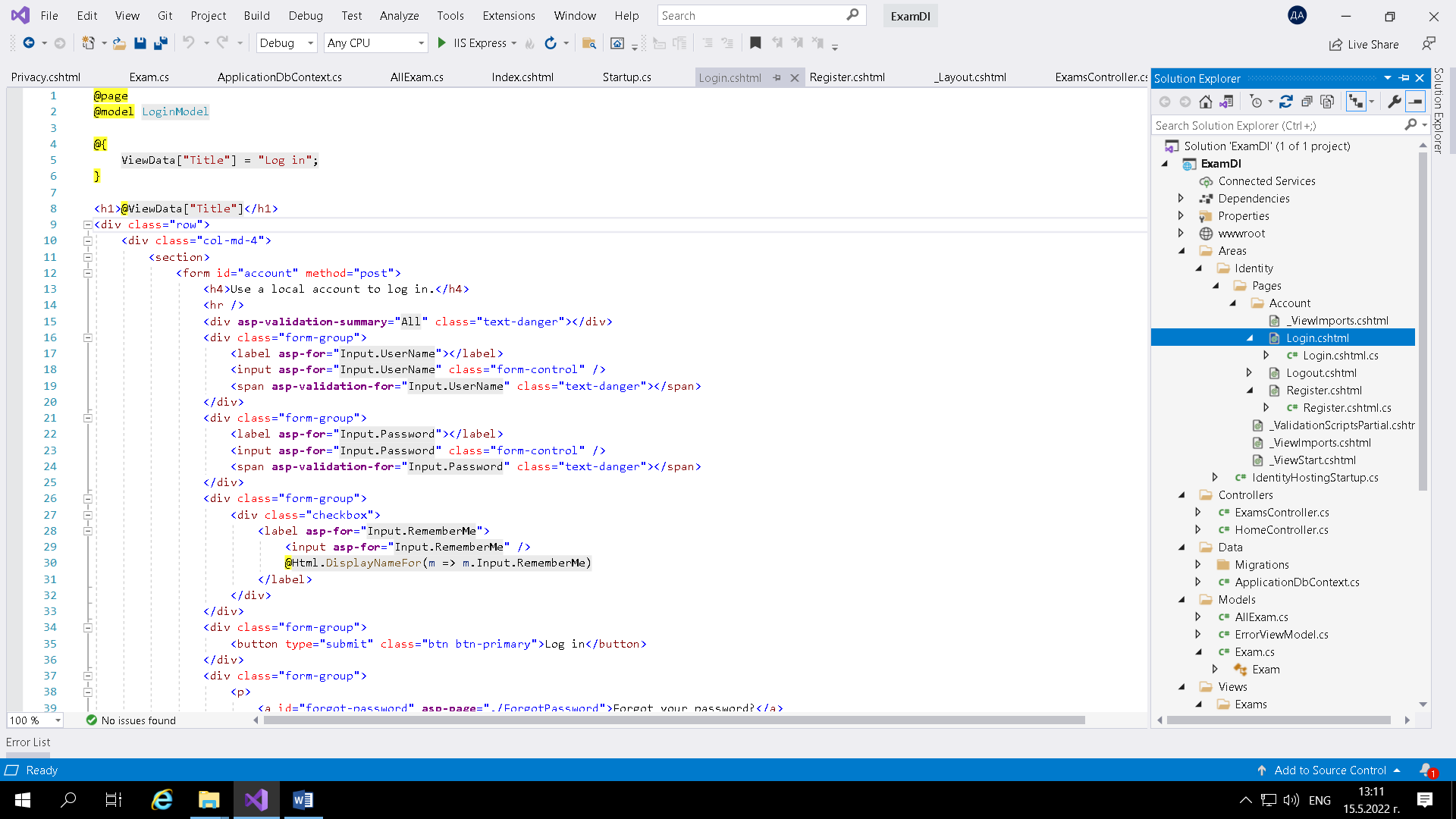
nav {

background-color: lightpink;

}

</style>

1. Автентикацията в Login с username I password



public class InputModel

{

[Required]

public string UserName { get; set; }

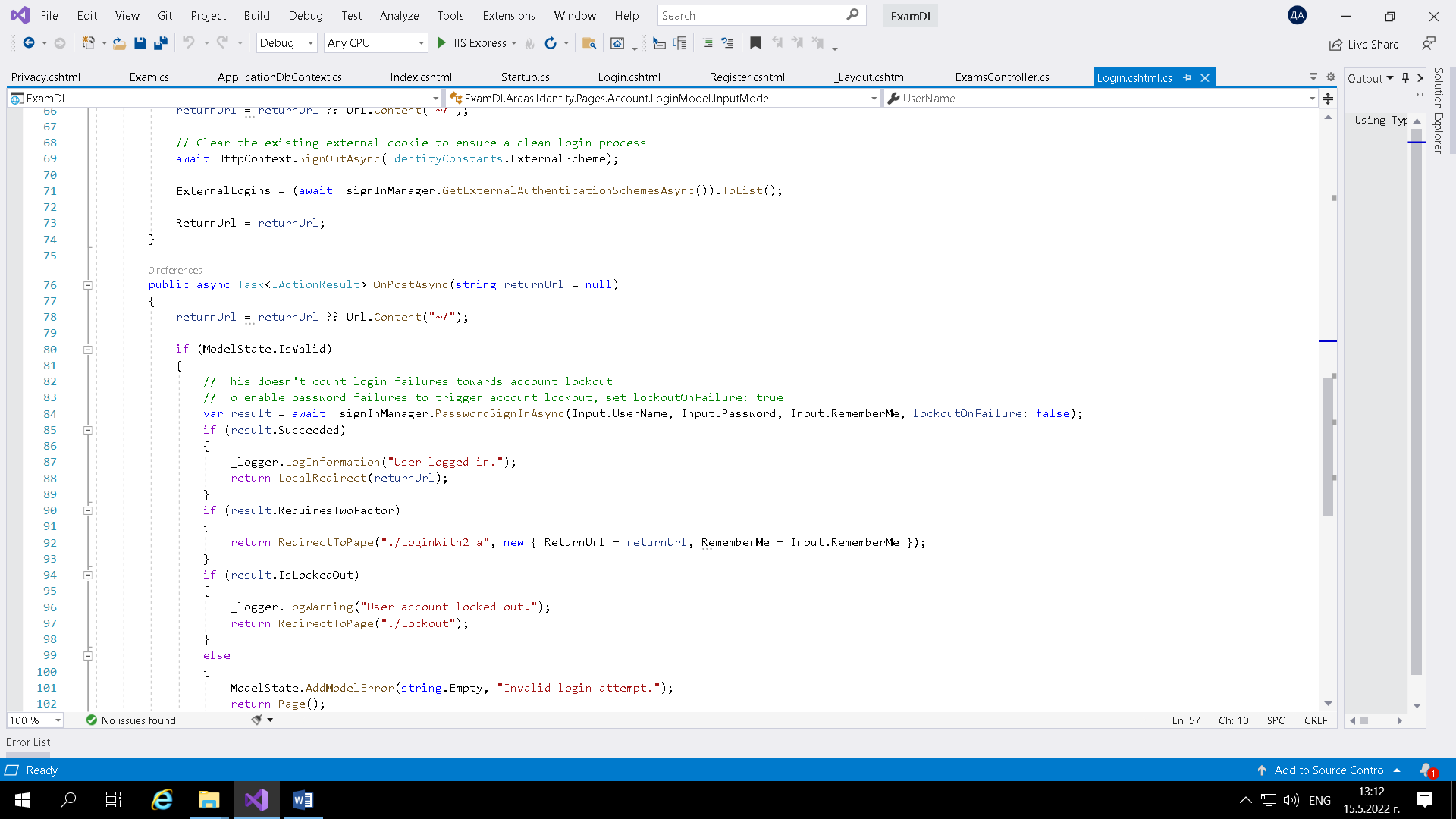
[Required]

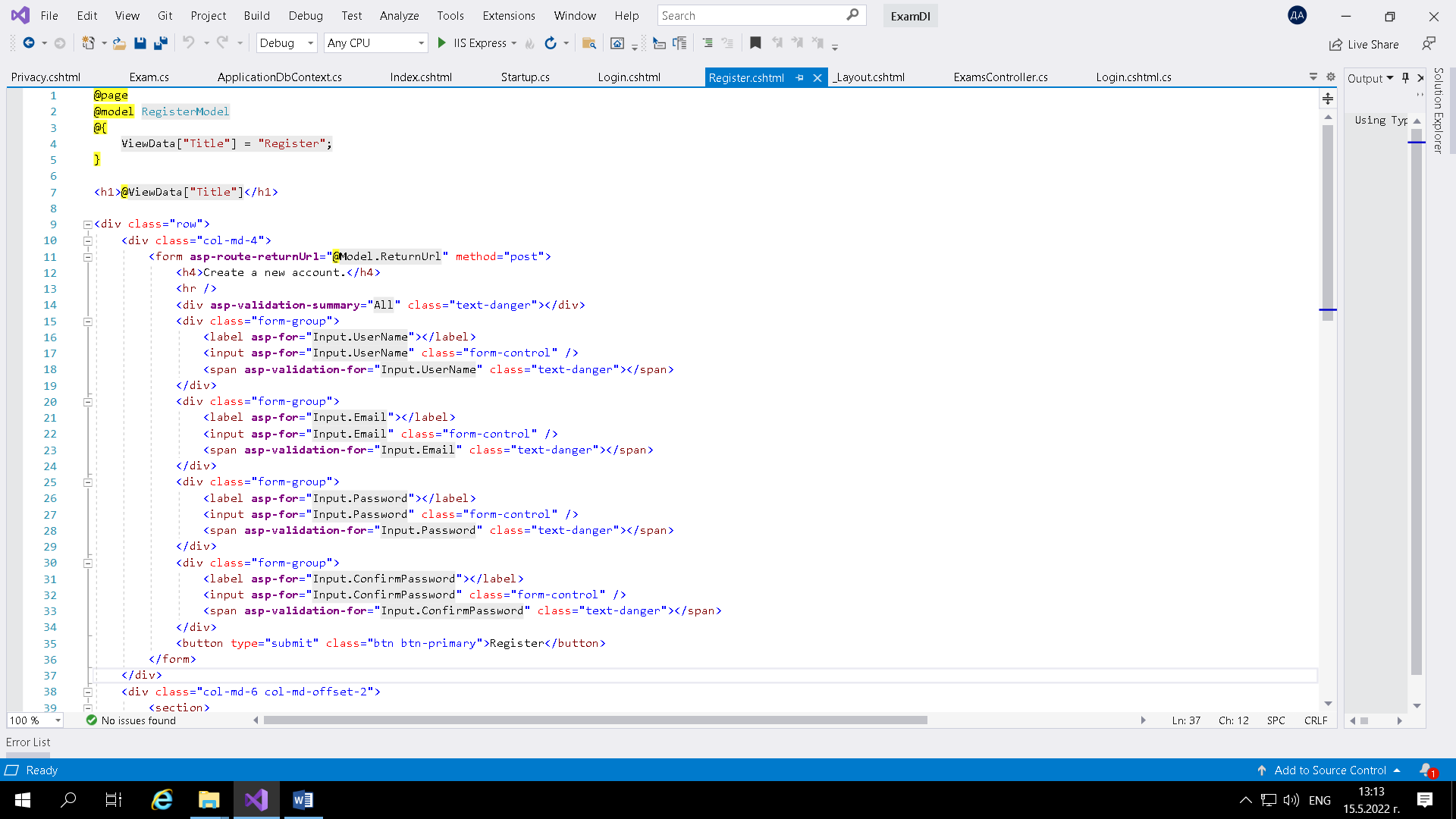
[DataType(DataType.Password)]

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; }

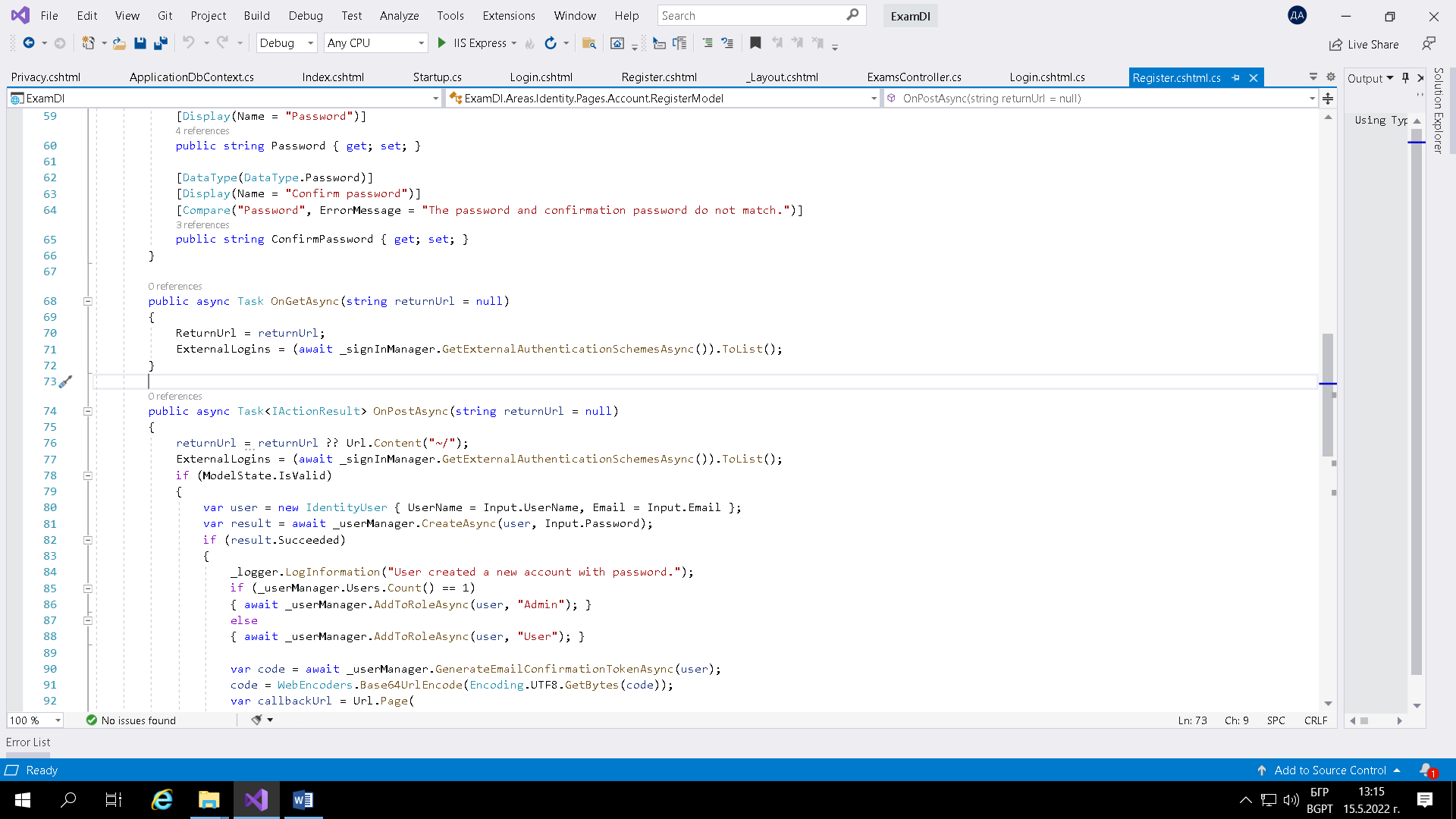
[DataType(DataType.Password)]

[Display(Name = "Confirm password")]

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

public string ConfirmPassword { get; set; }

}



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 \_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>

vi

{

<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>