

Сучасні інтернет технології. Лабораторне заняття №4
КОНФІГУРАЦІЯ ПРОЄКТУ ЗАСТОСУНКУ ASP.NET CORE

Виконала: студентка МІТ-41, Панченко Владислава

Завдання для виконання (max - 100 балів)

1. Забезпечте автентифікацію користувачів у застосунку. Реалізуйте сторінку входу, реєстрації та виходу, використовуючи ASP.NET Core Identity.

Перевірте, що неавтентифіковані користувачі не мають доступу до жодної сторінки, крім головної сторінки та сторінки реєстрації/автентифікації. (max - 15 балів)

```
builder.Services.AddControllersWithViews(options =>
{
    var policy = new AuthorizationPolicyBuilder()
        .RequireAuthenticatedUser()
        .Build();
    options.Filters.Add(new AuthorizeFilter(policy));
});
```

```
[AllowAnonymous]
public async Task<IActionResult> IndexAsync()
{
    var users = await _repository.ReadAll<WebApplicationUser>().ToListAsync();
    return View(users);
}

[AllowAnonymous]
public IActionResult Privacy()
{
    return View();
}
```

2. Створіть політику авторизації, яка дозволяє доступ до сторінки «Архів матеріалів» лише тим користувачам, які мають твердження IsVerifiedClient. Додайте твердження вручну під час реєстрації. (max - 15 балів)

```
builder.Services.AddAuthorization(options =>
{
    // Л4_2
    options.AddPolicy("ArchivePolicy", policy =>
        policy.RequireClaim("IsVerifiedClient"));
```

```
if (result.Succeeded)
{
    await _userManager.AddClaimAsync(user, new System.Security.Claims.Claim("IsVerifiedClient", "true"));
    await _userManager.AddClaimAsync(user, new System.Security.Claims.Claim("WorkingHours", "1"));
    await _userManager.AddClaimAsync(user, new System.Security.Claims.Claim("IsMentor", "true"));
```

```
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;

namespace WebApplication1.Controllers
{
    [Authorize(Policy = "ArchivePolicy")]
    0 references
    public class ArchiveController : Controller
    {
        0 references
        public IActionResult Index()
        {
            return Content("Ласкаво просимо до Архіву! Тільки для Verified Clients.");
        }
    }
}
```

3. Реалізуйте ресурсну авторизацію для сторінки редагування ресурсу. Кожен ресурс має автора, і лише автор може редагувати його. Використайте IAuthorizationService та створіть обробник, який перевіряє, чи поточний користувач є автором ресурсу. (max - 25 балів)

```
using System.ComponentModel.DataAnnotations.Schema;

namespace WebApplicationData.Data
{
    14 references
    public class AppResource
    {
        6 references
        public int Id { get; set; }
        7 references
        public string Title { get; set; }
        3 references
        public string AuthorId { get; set; }

        [ForeignKey("AuthorId")]
        4 references
        public WebApplicationUser Author { get; set; }
    }
}
```

```
using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace WebApplicationData.Data
{
    9 references
    public class WebApplicationDbContext : IdentityDbContext<WebApplicationUser>
    {
        0 references
        public WebApplicationDbContext(DbContextOptions<WebApplicationDbContext> options)
            : base(options)
        {
        }
        0 references
        public DbSet<AppResource> AppResources { get; set; }
    }
}
```

```
using Microsoft.AspNetCore.Authorization;

3 references
public class IsAuthorRequirement : IAuthorizationRequirement
{
```

```
using Microsoft.AspNetCore.Authorization;
using WebApplicationData.Data;
using System.Security.Claims;

1 reference
public class IsAuthorHandler : AuthorizationHandler<IsAuthorRequirement, AppResource>
{
    0 references
    protected override Task HandleRequirementAsync(
        AuthorizationHandlerContext context,
        IsAuthorRequirement requirement,
        AppResource resource)
    {
        var userId = context.User.FindFirst(ClaimTypes.NameIdentifier)?.Value;

        if (userId != null && resource.AuthorId == userId)
        {
            context.Succeed(requirement);
        }

        return Task.CompletedTask;
    }
}
```

```
builder.Services.AddSingleton<IAuthorizationHandler, IsAuthorHandler>();
```

```
[HttpPost]
0 references
public async Task<IActionResult> Edit(AppResource model)
{
    var resource = await _repository.SingleAsync<AppResource>(r => r.Id == model.Id);

    if (resource == null) return NotFound();

    var authorizationResult = await _authorizationService
        .AuthorizeAsync(User, resource, "ResourceOwner");

    if (!authorizationResult.Succeeded)
    {
        return Forbid();
    }

    resource.Title = model.Title;

    await _repository.UpdateAsync(resource);

    return RedirectToAction("Index");
}
```

4. Створіть кастомну вимогу авторизації MinimumWorkingHoursRequirement, яка дозволяє доступ до сторінки «Преміум» лише тим користувачам, які мають твердження WorkingHours з числовим значенням не менше 100. Реалізуйте обробник, який виконує перевірку. (max - 15 балів)

```
using Microsoft.AspNetCore.Authorization;

namespace WebApplication1.Authorization
{

    4 references
    public class MinHoursRequirement : IAuthorizationRequirement
    {
        2 references
        public int MinHours { get; }

        1 reference
        public MinHoursRequirement(int minHours)
        {
            MinHours = minHours;
        }
    }
}
```

```
using Microsoft.AspNetCore.Authorization;

namespace WebApplication1.Authorization
{
    public class MinHoursHandler : AuthorizationHandler<MinHoursRequirement>
    {
        protected override Task HandleRequirementAsync(AuthorizationHandlerContext context, MinHoursRequirement requirement)
        {
            var hoursClaim = context.User.FindFirst("WorkingHours");

            if (hoursClaim != null && int.TryParse(hoursClaim.Value, out int hours))
            {
                if (hours >= requirement.MinHours)
                {
                    context.Succeed(requirement);
                }
            }

            return Task.CompletedTask;
        }
    }
}
```

```
// ЛАЧ 4
options.AddPolicy("PremiumContent", policy =>
    policy.Requirements.Add(new MinHoursRequirement(100)));
```

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Authorization;

namespace WebApplication1.Controllers
{
    [Authorize(Policy = "PremiumContent")]
    public class PremiumController : Controller
    {
        public IActionResult Index()
        {
            return Content("Вітаємо! Ви маєте достатньо робочих годин (100+), щоб бачити цю преміум-сторінку.");
        }
    }
}
```

```
await _userManager.AddClaimAsync(user, new System.Security.Claims.Claim("WorkingHours", "1"));
```

5. Створіть політику, яка дозволяє доступ до сторінки «Форум» лише тим користувачам, які мають хоча б одне з тверджень: IsMentor, IsVerifiedUser, або HasForumAccess. Реалізуйте обробник, який перевіряє хоча б одне з цих тверджень. (max - 15 балів)

```
using Microsoft.AspNetCore.Authorization;

namespace WebApplication1.Authorization
{
    3 references
    public class ForumAccessRequirement : IAuthorizationRequirement
    {
    }
}
```

```
using Microsoft.AspNetCore.Authorization;
using System.Threading.Tasks;

namespace WebApplication1.Authorization
{
    1 reference
    public class ForumAccessHandler : AuthorizationHandler<ForumAccessRequirement>
    {
        0 references
        protected override Task HandleRequirementAsync(AuthorizationHandlerContext context, ForumAccessRequirement requirement)
        {
            bool hasAccess = context.User.HasClaim(c => c.Type == "IsMentor") ||
                            context.User.HasClaim(c => c.Type == "IsVerifiedUser") ||
                            context.User.HasClaim(c => c.Type == "HasForumAccess");

            if (hasAccess)
            {
                context.Succeed(requirement);
            }

            return Task.CompletedTask;
        }
    }
}
```

```
// Л4 5
options.AddPolicy("ForumPolicy", policy =>
    policy.Requirements.Add(new ForumAccessRequirement()));
```

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Authorization;

namespace WebApplication1.Controllers
{
    [Authorize(Policy = "ForumPolicy")]
    0 references
    public class ForumController : Controller
    {
        0 references
        public IActionResult Index()
        {
            return Content("Ласкаво просимо на Форум! Ви маєте одне з необхідних прав доступу.");
        }
    }
}
```

```
await _userManager.AddClaimAsync(user, new System.Security.Claims.Claim("IsMentor", "true"));
```

6. Зафіксувати зміни у проєкті на GitHub. (max - 10 балів)
7. Оформити звіт. (max - 5 балів)