Лабораторна робота №5

Звіт

З дисципліни “Сучасні інтернет технології”
на тему: “Глобалізація та локалізація застосунку ASP.NET Core ”.

Студента 4 курсу: Групи МІТ-41
Демиденко Андрій

Київ - 2025р.

Хід виконання роботи

На тему: «Глобалізація та локалізація застосунку ASP.NET Core»
Мета роботи: Здобути практичні навички впровадження багатомовності у веб-застосунок ASP.NET Core. Навчитися налаштовувати служби локалізації, створювати файли ресурсів, реалізувати перемикання мов та адаптувати формати даних (дат, чисел, валют) під різні культурні середовища .

Налаштування служб локалізації

У файл Program.cs було додано підтримку локалізації. Використано метод AddLocalization із зазначенням шляху до папки ресурсів (Resources). Також оновлено сервіси MVC методами AddViewLocalization та AddDataAnnotationsLocalization для підтримки перекладу у Views та валідації моделей .

```
+ // Configure Localization
+ builder.Services.AddLocalization(options => options.ResourcesPath = "Resources");
+
+ builder.Services.AddControllersWithViews()
+     .AddViewLocalization()
+     .AddDataAnnotationsLocalization();
+
+
```

Рис. 1 - код з Program.cs з доданими сервісами локалізації

Створення файлів ресурсів (.resx)

У проєкті створено папку Resources, яка дзеркально відображає структуру контролерів та представлень. Створено файли ресурсів для трьох культур: базової (англійська), української (uk-UA) та французької (fr-FR) (рис. 2).

- Для контролера: Resources/Controllers/HomeController.xx.resx.
- Для представлення: Resources/Views/Home/Index.xx.resx. У файли додано ключі WelcomeMessage та Title з відповідними перекладами .

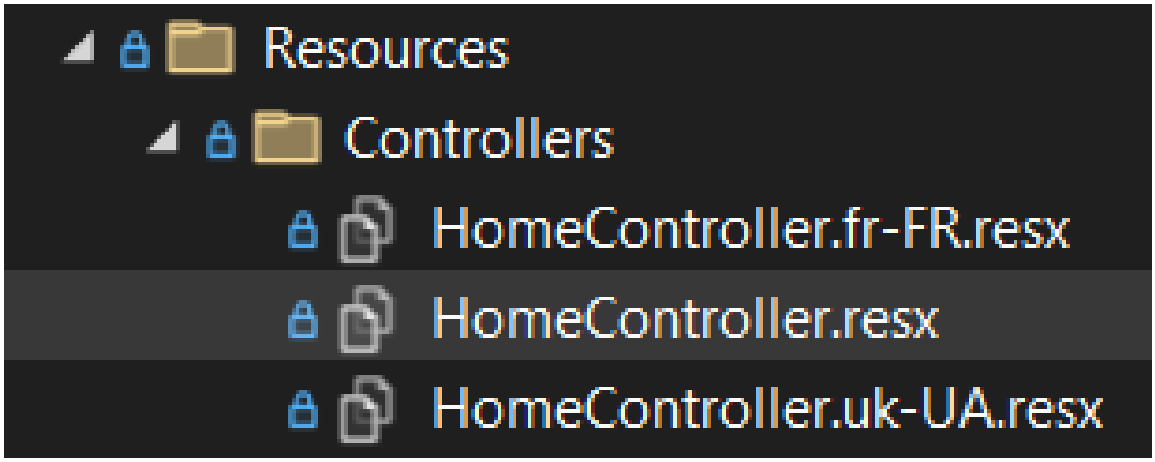

Рис. 2 - Solution Explorer зі структурою папки Resources.

Налаштування RequestLocalizationOptions

У Program.cs налаштовано параметри локалізації запиту. Визначено список підтримуваних культур (en-US, uk-UA, fr-FR) та встановлено культуру за замовчуванням (en-US). Ці налаштування застосовано як до форматування (SupportedCultures), так і до ресурсів інтерфейсу (SupportedUICultures) (рис. 3).

```
+ // define the list of supported cultures
+ var supportedCultures = new[] { "en-US", "uk-UA", "fr-FR" };
+
+ builder.Services.Configure<RequestLocalizationOptions>(options =>
+ {
+     options.SetDefaultCulture("en-US");
+
+     options.AddSupportedCultures(supportedCultures);
+
+     options.AddSupportedUICultures(supportedCultures);
+ });
+
```

Рис. 3 - код конфігурації RequestLocalizationOptions у Program.cs.

Додавання Middleware локалізації

До конвеєра обробки запитів додано app.UseRequestLocalization(). Цей компонент розміщено **перед** UseRouting та UseAuthorization, щоб система могла визначити культуру користувача до початку обробки логіки запиту (рис. 4).

```
163     app.UseHttpsRedirection();
164     app.UseStaticFiles();
165
166   + app.UseRequestLocalization();
167   +
168     app.UseRouting();
169
170     app.UseAuthentication();
```

Рис. 4 - частина Program.cs з підключенням middleware.

Реалізація типізованої локалізації

У контролері: Впроваджено сервіс IStringLocalizer<HomeController> через конструктор. Отримане значення локалізованого рядка (_localizer["WelcomeMessage"]) передається у ViewData.

У представленні: Впроваджено сервіс IViewLocalizer. Використано його для локалізації заголовка сторінки (@Localizer["Title"]) та відображення даних з контролера (рис. 5).

```
16  +       private readonly IStringLocalizer<HomeController> _localizer;
17
18  +       public HomeController(IWebAppRepository repository, WebAppConfiguration config,
        IStringLocalizer<HomeController> localizer)
19          {
20              _repository = repository;
21              _config = config;
22  +           _localizer = localizer;
23          }
24  +
25          [AllowAnonymous]
26          public Task<IActionResult> Index()
27          {
28              var users = _repository.All<ApplicationUser>();
29
30              ViewData["ApplicationName"] = _config.ApplicationName;
31              ViewData["ApiUrl"] = _config.MySpecificSetting?.ApiUrl;
32  +           ViewData["WelcomeMessage"] = _localizer["WelcomeMessage"];
33
34              return Task.FromResult<IActionResult>(View(users));
35  +       }
```

```
1   + @using Microsoft.AspNetCore.Mvc.Localization
2   + @inject IViewLocalizer Localizer
3   +
4   + @{
5   +     ViewData["Title"] = Localizer["Title"];
6     }
7
8     <div class="text-center">
9   +     <h1 class="display-4">@Localizer["Title"]</h1>
10  +


11  +     <p class="lead">@ViewData["WelcomeMessage"]</p>
12  +
13  +     <p>Learn about <a href="https://docs.microsoft.com/aspnet/core">building Web apps with ASP.NET
        Core</a>.</p>
14  + </div>
```

Рис. 5 - код HomeController.cs та Index.cshtml з використанням локалізаторів.

Реалізація перемикання мов

Створено контролер SelectLanguageController з методом SetLanguage, який приймає код культури, записує його у Cookie (.AspNetCore.Culture) і повертає користувача на попередню сторінку . Створено часткове представлення _SelectLanguagePartial.cshtml з формою та випадаючим списком мов, яке інтегровано в навігаційну панель _Layout.cshtml (рис. 6) .

```
@using Microsoft.AspNetCore.Builder
@using Microsoft.AspNetCore.Localization
@using Microsoft.AspNetCore.Mvc.Localization
@using Microsoft.Extensions.Options

@inject IViewLocalizer Localizer
@inject IOptions<RequestLocalizationOptions> LocOptions

@{
    var requestCulture = Context.Features.Get<IRequestCultureFeature>();
    var cultureItems = LocOptions.Value.SupportedUICultures
        .Select(c => new SelectListItem { Value = c.Name, Text = c.DisplayName })
        .ToList();

    var returnUrl = string.IsNullOrEmpty(Context.Request.Path) ? "~/" : $"{Context.Request.Path.Value}{Context.Request.QueryString}";
}

<div title="@Localizer["Request culture provider:"] @requestCulture?.Provider?.GetType().Name">
    <form id="selectLanguage" asp-controller="SelectLanguage"
          asp-action="SetLanguage" asp-route-returnUrl="@returnUrl"
          method="post" class="form-horizontal" role="form">

        <label asp-for="@requestCulture.RequestCulture.UICulture.Name">@Localizer["Language:"]</label>

        <select name="culture"
                onchange="this.form.submit();"
                asp-for="@requestCulture.RequestCulture.UICulture.Name"
                asp-items="cultureItems"
                class="form-select d-inline w-auto">
        </select>
    </form>
</div>
```

Рис. 6 - інтерфейс сайту з випадаючим списком мов у меню.

Локалізація форматів даних

На головній сторінці реалізовано блок тестування, що виводить поточну дату, число та валюту. Продемонстровано, що при зміні мови автоматично змінюються формати (рис. 7):

- **uk-UA:** дата dd.MM.yyyy, роздільник коми, валюта ₴.
- **en-US:** дата MM/dd/yyyy, роздільник крапки, валюта $.
- **fr-FR:** дата dd/MM/yyyy, валюта € .

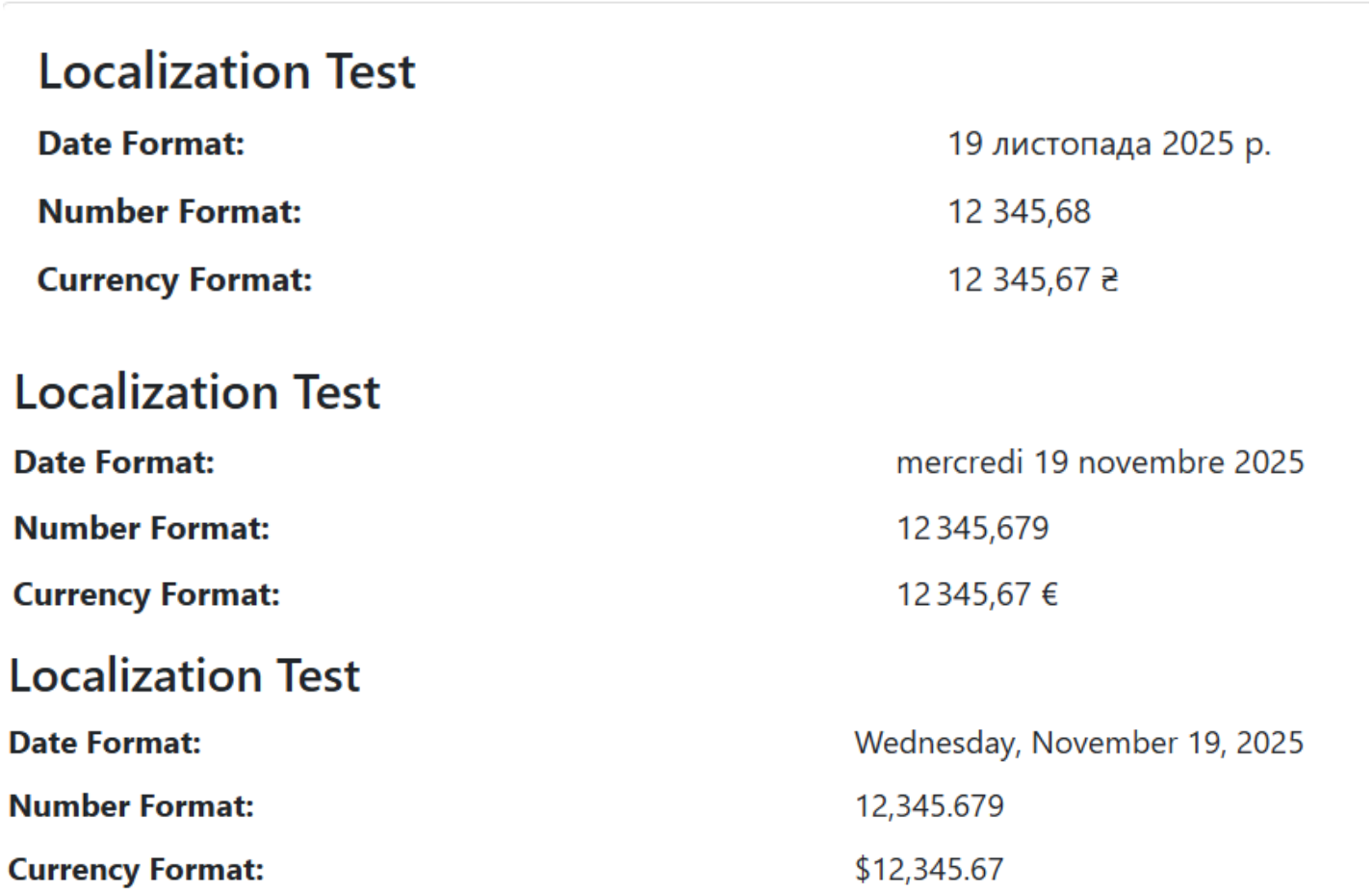

Рис. 7 - французькою/англійською/українською мовою.

Висновок

У ході виконання лабораторної роботи було успішно реалізовано глобалізацію та локалізацію веб-застосунку ASP.NET Core. Ми налаштували інфраструктуру локалізації, створили словники ресурсів для трьох мов та реалізували механізм вибору мови через Cookies та параметри URL. Також було підтверджено, що обрана культура коректно впливає як на переклад текстового контенту, так і на формати відображення дат і чисел.