**ЛАБОРАТОРНА РОБОТА № 3**

**Початок роботи з ASP.NET Core 5. Middleware**

***Мета***: ознайомитися з файловими та нефайловими провайдерами конфігурації в ASP.NET, принципами побудови власних провайдерів логування, визначенням маршрутів, навчитися зберігати дані у сесіях.

**Хід роботи:**

**Завдання 2:** Створити у Visual Studio **ASP.NET Core Empty**.

Назва рішення: **AspDotNetLabs**

Назва проекту: **AspDotNetLab3**

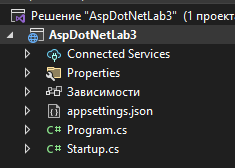


Рис. 2.0.0 Оглядач рішень з відкритим проектом

**Частина 1.** Створити власний json-файл конфігурації з назвою conf.json. У файлі конфігурації розмістити наступні налаштування:

● Title - назва, яка буде виводитися при зверненні до будь-якого маршруту додатку

● LogFile - шлях (назва) текстового файла для запису логів.

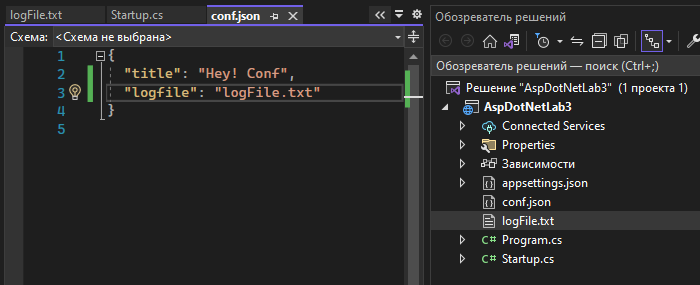
****

Рис. 2.1.1 Створений власний json-файл конфігурації з назвою conf.json

**Частина 2:** За допомогою MapRoute визначити маршрути та їх обробники для того, щоб працювали наступні маршрути:

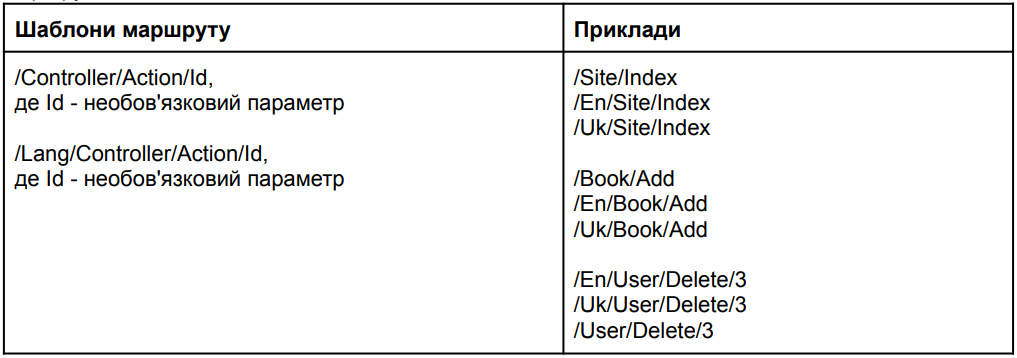


Рис. 2.2.1. Приклад шаблонів маршрутів, які мають оброблятися

var routeBuilder = new RouteBuilder(app);

routeBuilder.MapRoute("/{controller}/{action}/{id:int?}",

async context => {

context.Response.ContentType = "text/html; charset=utf-8";

await context.Response.WriteAsync($"Route \"<b>{context.Request.Path}</b>\" is valid");

});

routeBuilder.MapRoute("/{lang}/{controller}/{action}/{id:int?}",

async context => {

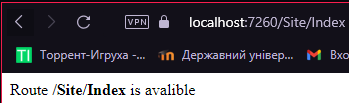
context.Response.ContentType = "text/html; charset=utf-8";

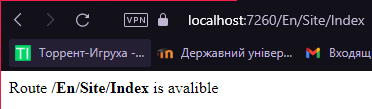
await context.Response.WriteAsync($"Route \"<b>{context.Request.Path}</b>\" is valid");

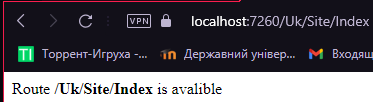
});

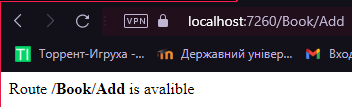
app.UseRouter(routeBuilder.Build());

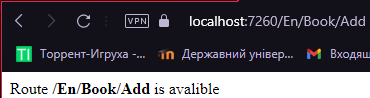
Результат виконання:

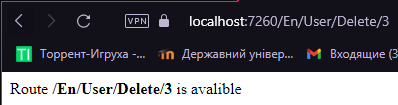












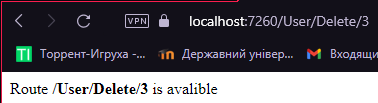
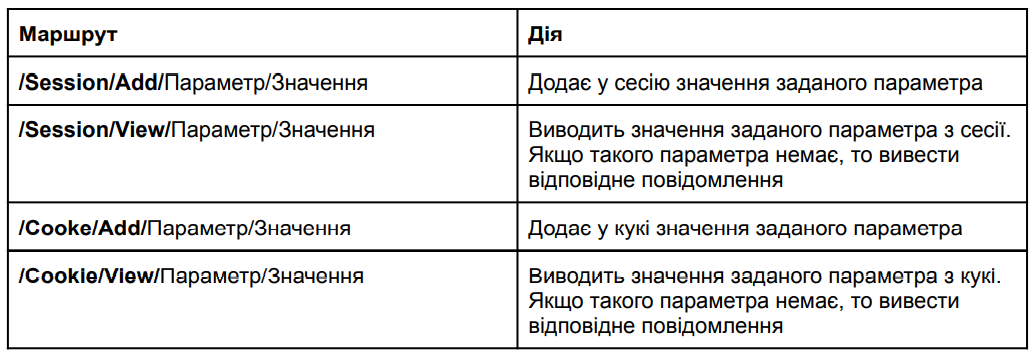


Рис. 2.2.2- 2.2.7. Результат обробки маршрутів

**Частина 3:** Визначити маршрути та реалізувати вказані дії при переході за ними:



Лістинг коду ConfigureServices:

public void ConfigureServices(IServiceCollection services)

{

services.AddDistributedMemoryCache();

services.AddSession(options =>

{

options.IdleTimeout = TimeSpan.FromMinutes(30);

options.Cookie.HttpOnly = true;

options.Cookie.IsEssential = true;

});

}

Лістинг коду Startup.cs:

routeBuilder.MapRoute("/session/add/{key}/{value}",

async context =>

{

var key = $"{context.Request.Path}".Split("/")[3];

var value = $"{context.Request.Path}".Split("/")[4];

context.Session.SetString(key, value);

await context.Response.WriteAsync($"Key with value added to the session");

});

routeBuilder.MapRoute("/session/view/{key}",

async context =>

{

var key = $"{context.Request.Path}".Split("/")[3];

var value = context.Session.GetString(key);

if (value != null)

{

await context.Response.WriteAsync($"<b>{key}</b> (key) = {value}(value)");

return;

}

await context.Response.WriteAsync($"No value was found for this key in the session");

});

routeBuilder.MapRoute("/cookie/add/{key}/{value}",

async context =>

{

var key = $"{context.Request.Path}".Split("/")[3];

var value = $"{context.Request.Path}".Split("/")[4];

context.Response.Cookies.Append(key, value);

await context.Response.WriteAsync($"Key with value added to the cookies");

}

);

routeBuilder.MapRoute("/cookie/view/{key}",

async context =>

{

var key = $"{context.Request.Path}".Split("/")[3];

if (context.Request.Cookies.ContainsKey(key))

{

var value = context.Request.Cookies[key];

await context.Response.WriteAsync($"<b>{key}</b> (key) = {value} (value)");

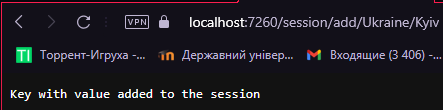
return;

}

await context.Response.WriteAsync($"No value was found for this key");

});

app.UseRouter(routeBuilder.Build());



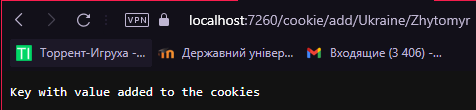
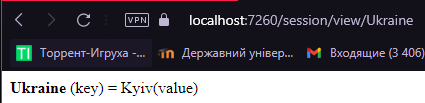


Рис. 2.3.1- 2.3.2. Додавання значення до сесії та кукі.



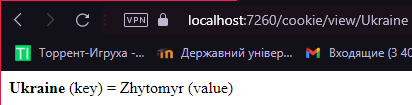
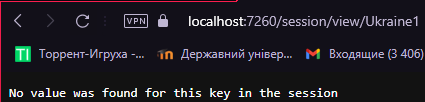


Рис. 2.3.3- 2.3.4.Діставання значення з сесії та кукі.



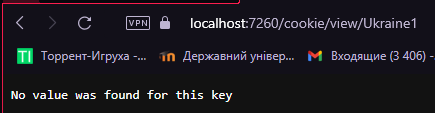


Рис. 2.3.5- 2.3.6.Спроба діставання неіснуючого значення з сесії та кукі.

**Частина 4:** Створити власний файловий провайдер логування, який буде записувати інформацію про дату, час, IP-адресу користувача та шлях запиту до сервера в текстовий файл, назва якого задана в конфігурації в параметрі LogFile. При кожному запиті має додавати один рядок до файла.

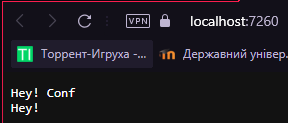


Рис. 2.4.1. Результат виконная програми (відображення при кодному маршруті привітання, яке знаходиться в конф. файлі)

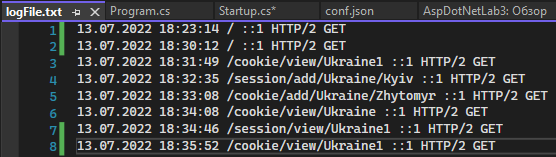


Рис. 2.4.2. Вміст файлу log.txt

**Завдання 3.**

Запушити виконану роботу у репозиторій на GitHub за назвою AspDotNetLab1 і відкрити доступ для викладачів:

● [morozov@ztu.edu.ua](mailto:morozov@ztu.edu.ua)

● [4ov.ztu@gmail.com](mailto:4ov.ztu@gmail.com)

Посилання на репозиторій:

https://github.com/IvanPaliy/AspDotNetLab3.git

***Висновки:*** в ході виконання лабораторної роботи я ознайомився з файловими та нефайловими провайдерами конфігурації в ASP.NET, принципами побудови власних провайдерів логування, визначенням маршрутів, навчитися зберігати дані у сесіях.