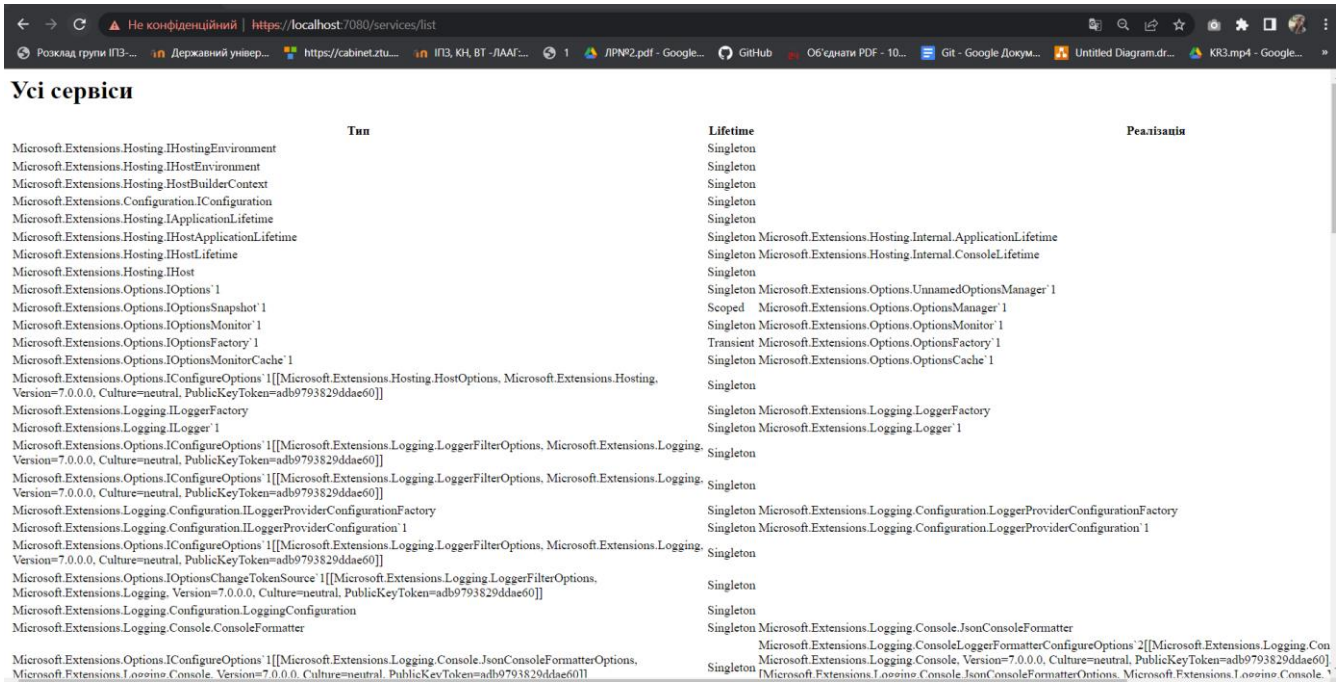
ЛАБОРАТОРНА РОБОТА № 2

Тема: Сервіси та Dependency Injection

Мета роботи: ознайомитися зі структурою ASP.NET Code додатків, навчитися створювати найпростіші та використовувати наявні компоненти middleware у ASP.NET Core.

Хід роботи:

1. Зробити обробку звернення до адреси “/services/list”, реалізувавши виведення інформації про всі сервіси, додані у додаток (тип сервісу, тип реалізації сервісу, життєвий цикл сервісу). У звіті до лабораторної роботи навести коротку інформацію про кожний сервіс (достатньо одного речення, що чітко описує сервіс).



Усі сервіси

Тип	Lifetime	Реалізація
Microsoft.Extensions.Hosting.IHostingEnvironment	Singleton	
Microsoft.Extensions.Hosting.IHostEnvironment	Singleton	
Microsoft.Extensions.Hosting.HostBuilderContext	Singleton	
Microsoft.Extensions.Configuration.IConfiguration	Singleton	
Microsoft.Extensions.Hosting.IApplicationLifetime	Singleton	
Microsoft.Extensions.Hosting.IHostApplicationLifetime	Singleton	Microsoft.Extensions.Hosting.Internal.ApplicationLifetime
Microsoft.Extensions.Hosting.IHostLifetime	Singleton	Microsoft.Extensions.Hosting.Internal.ConsoleLifetime
Microsoft.Extensions.Hosting.IHost	Singleton	
Microsoft.Extensions.Options.IOptions`1	Singleton	Microsoft.Extensions.Options.UnnamedOptionsManager`1
Microsoft.Extensions.Options.IOptionsSnapshot`1	Scoped	Microsoft.Extensions.Options.OptionsManager`1
Microsoft.Extensions.Options.IOptionsMonitor`1	Singleton	Microsoft.Extensions.Options.OptionsMonitor`1
Microsoft.Extensions.Options.IOptionsFactory`1	Transient	Microsoft.Extensions.Options.OptionsFactory`1
Microsoft.Extensions.Options.IOptionsMonitorCache`1	Singleton	Microsoft.Extensions.Options.OptionsCache`1
Microsoft.Extensions.Options.IConfigureOptions`1[[Microsoft.Extensions.Hosting.HostOptions, Microsoft.Extensions.Hosting, Version=7.0.0.0, Culture=neutral, PublicKeyToken=adb9793829ddae60]]	Singleton	
Microsoft.Extensions.Logging.ILoggerFactory	Singleton	Microsoft.Extensions.Logging.LoggerFactory
Microsoft.Extensions.Logging.ILogger`1	Singleton	Microsoft.Extensions.Logging.Logger`1
Microsoft.Extensions.Options.IConfigureOptions`1[[Microsoft.Extensions.Logging.LoggerFilterOptions, Microsoft.Extensions.Logging, Version=7.0.0.0, Culture=neutral, PublicKeyToken=adb9793829ddae60]]	Singleton	
Microsoft.Extensions.Options.IConfigureOptions`1[[Microsoft.Extensions.Logging.LoggerFilterOptions, Microsoft.Extensions.Logging, Version=7.0.0.0, Culture=neutral, PublicKeyToken=adb9793829ddae60]]	Singleton	
Microsoft.Extensions.Logging.Configuration.ILoggerProviderConfigurationFactory	Singleton	Microsoft.Extensions.Logging.Configuration.LoggerProviderConfigurationFactory
Microsoft.Extensions.Logging.Configuration.ILoggerProviderConfiguration`1	Singleton	Microsoft.Extensions.Logging.Configuration.LoggerProviderConfiguration`1
Microsoft.Extensions.Options.IConfigureOptions`1[[Microsoft.Extensions.Logging.LoggerFilterOptions, Microsoft.Extensions.Logging, Version=7.0.0.0, Culture=neutral, PublicKeyToken=adb9793829ddae60]]	Singleton	
Microsoft.Extensions.Options.IOptionsChangeTokenSource`1[[Microsoft.Extensions.Logging.LoggerFilterOptions, Microsoft.Extensions.Logging, Version=7.0.0.0, Culture=neutral, PublicKeyToken=adb9793829ddae60]]	Singleton	
Microsoft.Extensions.Logging.Configuration.LoggingConfiguration	Singleton	
Microsoft.Extensions.Logging.Console.ConsoleFormatter	Singleton	Microsoft.Extensions.Logging.Console.JsonConsoleFormatter
Microsoft.Extensions.Options.IConfigureOptions`1[[Microsoft.Extensions.Logging.Console.JsonConsoleFormatterOptions, Microsoft.Extensions.Logging.Console, Version=7.0.0.0, Culture=neutral, PublicKeyToken=adb9793829ddae60]]	Singleton	Microsoft.Extensions.Logging.ConsoleLoggerFormatterConfigureOptions`2[[Microsoft.Extensions.Logging.Con Microsoft.Extensions.Logging.Console, Version=7.0.0.0, Culture=neutral, PublicKeyToken=adb9793829ddae60] [Microsoft.Extensions.Logging.Console.JsonConsoleFormatterOptions, Microsoft.Extensions.Logging.Console, `

Рис.1.

Опис серверів:

- Microsoft.Extensions.Hosting.IHostingEnvironment: цей сервіс містить інформацію про поточне середовище виконання додатку.
- Microsoft.Extensions.Hosting.IHostEnvironment: цей сервіс містить інформацію про поточне середовище виконання додатку, як IHostingEnvironment, але рекомендується використовувати його замість IHostingEnvironment.
- Microsoft.Extensions.Hosting.HostBuilderContext: цей сервіс містить контекст для будівництва та конфігурації хоста додатку.

- Microsoft.Extensions.Configuration.IConfiguration: цей сервіс дозволяє додатку отримувати доступ до конфігураційних даних.
- Microsoft.Extensions.Hosting.IApplicationLifetime: цей сервіс дозволяє додатку керувати своїм життєвим циклом.
- Microsoft.Extensions.Hosting.IHostApplicationLifetime: цей сервіс містить інформацію про життєвий цикл хоста додатку.
- Microsoft.Extensions.Hosting.IHostLifetime: цей сервіс дозволяє додатку керувати життєвим циклом хоста.
- Microsoft.Extensions.Hosting.IHost: цей сервіс містить інформацію про хост додатку та дозволяє керувати його життєвим циклом.
- Microsoft.Extensions.Options.IOptions`1: цей сервіс дозволяє додатку отримувати налаштування з конфігурації.
- Microsoft.Extensions.Options.IOptionsSnapshot`1: цей сервіс дозволяє додатку отримувати оновлені налаштування з конфігурації під час роботи додатку.
- Microsoft.Extensions.Options.IOptionsMonitor`1: цей сервіс дозволяє додатку відстежувати та отримувати оновлені налаштування з конфігурації під час роботи додатку.
- Microsoft.Extensions.Options.IOptionsFactory`1: цей сервіс дозволяє додатку створювати об'єкти на основі налаштувань з конфігурації.
- Microsoft.Extensions.Options.IOptionsMonitorCache`1: цей сервіс дозволяє кешувати оновлені налаштування з конфігурації та додавати їх у список відстежуваних налаштувань.
- Microsoft.Extensions.Options.IOptionsChangeTokenSource<T>: Визначає джерело для маркерів змін, які використовуються для вказівки, коли параметри були змінені.
- Microsoft.Extensions.Logging.Configuration.LoggingConfiguration: Представляє конфігурацію журналювання.
- Microsoft.Extensions.Logging.Console.ConsoleFormatter: Форматує повідомлення журналу для виведення на консоль.
- Microsoft.Extensions.Options.IConfigureOptions<T>: Налаштовує певний тип параметрів.
- Microsoft.Extensions.Options.IOptionsChangeTokenSource<T>: Визначає джерело для маркерів змін, які використовуються для вказівки, коли параметри були змінені.
- Microsoft.Extensions.Logging.Console.ConsoleFormatter: Форматує повідомлення журналу для виведення на консоль.
- Microsoft.Extensions.Options.IConfigureOptions<T>: Налаштовує певний тип параметрів.

- Microsoft.Extensions.Options.IOptionsChangeTokenSource<T>: Визначає джерело для маркерів змін, які використовуються для вказівки, коли параметри були змінені.
- Microsoft.Extensions.Logging.Console.ConsoleFormatter: Форматує повідомлення журналу для виведення на консоль.
- Microsoft.Extensions.Options.IConfigureOptions<T>: Налаштовує певний тип параметрів.
- Microsoft.Extensions.Options.IOptionsChangeTokenSource<T>: Визначає джерело для маркерів змін, які використовуються для вказівки, коли параметри були змінені.
- Microsoft.Extensions.Logging.ILoggerProvider: Представляє тип, який надає екземпляри ILogger.
- Microsoft.Extensions.Options.IConfigureOptions and Microsoft.Extensions.Options.IOptionsChangeTokenSource це інтерфейси, які використовуються для налаштування параметрів у програмах .NET.
- Microsoft.Extensions.Logging.ILoggerProvider це інтерфейс, який використовується для входу в програми .NET.
- Microsoft.Extensions.Logging.EventSource.LoggingEventSource це джерело реєстрації подій, яке забезпечує інтерфейс на основі подій для реєстрації в програмах .NET.
- Microsoft.Extensions.Options.IConfigureOptions це інтерфейс, який використовується для налаштування параметрів у програмах .NET.
- Microsoft.AspNetCore.Hosting.IWebHostEnvironment, Microsoft.AspNetCore.Hosting.IHostingEnvironment, and Microsoft.AspNetCore.Hosting.IApplicationLifetime це інтерфейси, які використовуються для керування середовищем розміщення та терміном служби програми в ASP.NET Core.
- Microsoft.Extensions.Options.IConfigureOptions and Microsoft.Extensions.Options.IOptionsChangeTokenSource також використовуються для налаштування параметрів у ASP.NET Core.
- System.Diagnostics.DiagnosticListener, System.Diagnostics.DiagnosticSource, System.Diagnostics.ActivitySource, and System.Diagnostics.DistributedContextPropagator це класи, які використовуються для розподіленого трасування в програмах .NET.
- Microsoft.AspNetCore.Http.IHttpContextFactory, Microsoft.AspNetCore.Http.IMiddlewareFactory, Microsoft.AspNetCore.Hosting.Builder.IApplicationBuilderFactory, and Microsoft.AspNetCore.Hosting.IStartupFilter це інтерфейси, які використову-

ються для керування проміжним програмним забезпеченням і конвеєрами запитів у ASP.NET Core.

- Microsoft.AspNetCore.Routing.LinkParser - це сервіс, який дозволяє розбирати шаблони посилань для маршрутизації в ASP.NET Core.
- Microsoft.AspNetCore.Routing.Matching.EndpointSelector - це сервіс, який визначає найкращий кандидат для обробки HTTP-запиту на основі збігу з шаблоном маршруту та інших факторів.
- Microsoft.AspNetCore.Routing.MatcherPolicy - це базовий клас для сервісів, які впливають на відповідність маршруту.
- Microsoft.AspNetCore.Routing.Template.TemplateBinderFactory - це сервіс, який створює екземпляри TemplateBinder, який відповідає за виконання збігу шаблону маршруту з URL.
- Microsoft.AspNetCore.Routing.Patterns.RoutePatternTransformer - це сервіс, який дозволяє перетворювати шаблони маршруту з одного формату в інший.
- Microsoft.Extensions.Options.IConfigureOptions`1[[Microsoft.AspNetCore.Routing.RouteHandlerOptions, Microsoft.AspNetCore.Routing, Version=7.0.0.0, Culture=neutral, PublicKeyToken=adb9793829ddae60]] - це сервіс, який дозволяє настроювати параметри обробника маршрутів, такі як час очікування та обмеження обробки запиту.
- Microsoft.Extensions.Options.IConfigureOptions`1[[Microsoft.AspNetCore.Hosting.GenericWebHostServiceOptions, Microsoft.AspNetCore.Hosting, Version=7.0.0.0, Culture=neutral, PublicKeyToken=adb9793829ddae60]] - це сервіс, який дозволяє настроювати опції загального веб-хостингу, такі як шлях до файлу журналу.
- Microsoft.Extensions.Hosting.IHostedService - це базовий інтерфейс для всіх служб, які можуть бути запущені як фонові процеси в ASP.NET Core.

Лістинг програми:

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Http;
using Microsoft.Extensions.DependencyInjection;
using AspDotNetLab2;
using System.Text;

var builder = WebApplication.CreateBuilder(args);
var app = builder.Build();
var services = builder.Services;

app.MapGet("/", () => "Hello World!");
app.UseRouting();

app.MapGet("/services/list", async context =>
{
    var services = builder.Services;
```

```
    var middleware = new ServicesListMiddleware(null, services);
    await middleware.InvokeAsync(context);
});

app.Run();
```

***Лістинг** ServicesListMiddleware:*

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.Extensions.DependencyInjection;
using System.Linq;
using System.Text;

namespace AspDotNetLab2
{
    public class ServicesListMiddleware
    {
        private readonly RequestDelegate _next;
        private readonly IServiceCollection _services;

        public ServicesListMiddleware(RequestDelegate next, IServiceCollection services)
        {
            _next = next;
            _services = services;
        }

        public async Task InvokeAsync(HttpContext context)
        {
            var sb = new StringBuilder();
            sb.Append("<h1>Усі сервіси</h1><table>");
            sb.Append("<tr><th>Тип</th><th>Lifetime</th><th>Реалізація</th></tr>");
            foreach (var svc in _services)
            {
                sb.Append("<tr>");
                sb.Append($"<td>{svc.ServiceType.FullName}</td>");
                sb.Append($"<td>{svc.Lifetime}</td>");
                sb.Append($"<td>{svc.ImplementationType?.FullName}</td>");
                sb.Append("</tr>");
            }
            sb.Append("</table>");
            context.Response.ContentType = "text/html;charset=utf-8";
            await context.Response.WriteAsync(sb.ToString());
        }
    }
}
```

2. Створити три сервіси з різними життєвими циклами і продемонструвати їх роботу:

2.1. Тип “Transient”, назва “TimerService” - повертає поточну дату та час. Результати вивести за адресою “/services/timer”

Лістинг програми:

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Http;
using Microsoft.Extensions.DependencyInjection;
using AspDotNetLab2;
using System.Text;

var builder = WebApplication.CreateBuilder(args);
builder.Services.AddTransient<TimerService>();
```

```
var app = builder.Build();
var services = builder.Services;

app.MapGet("/", () => "Hello World!");
app.UseRouting();

app.MapGet("/services/list", async context =>
{
    var services = builder.Services;
    var middleware = new ServicesListMiddleware(null, services);
    await middleware.InvokeAsync(context);
});

app.MapGet("/services/timer", async context =>
{
    var timerService = app.Services.GetService<TimerService>();
    await context.Response.WriteAsync(
          $"Date and Time: {timerService?.GetCurrentDateTime()}");
});

app.Run();
```

Лістинг класу `TimerService`:

```
namespace AspDotNetLab2
{
    public class TimerService
    {
        public DateTime GetCurrentDateTime() => DateTime.Now;
    }
}
```

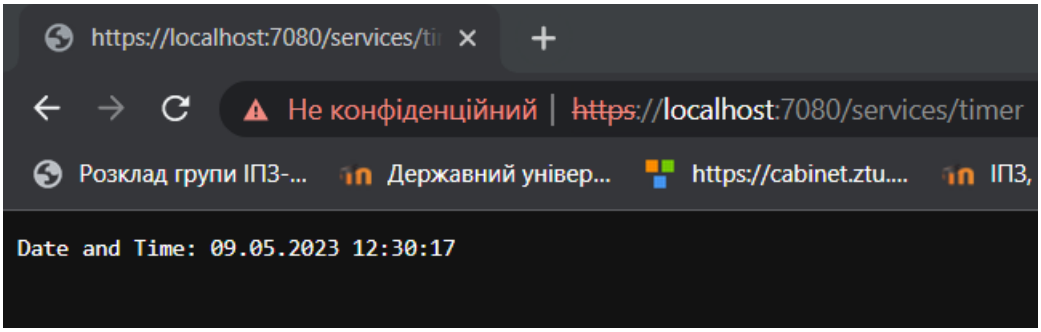


Рис.2.1. результат виконання програми

2.2. Тип “Scoped”, назва “RandomService” - в конструкторі генерує випадкове число. За допомогою методу чи властивості повертає його. Результати роботи вивести за адресою “/services/random”. Має виводитися два числа, які співпадають.

Лістинг програми:

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Http;
using Microsoft.Extensions.DependencyInjection;
using AspDotNetLab2;
using System.Text;

var builder = WebApplication.CreateBuilder(args);
builder.Services.AddTransient<TimerService>();
builder.Services.AddTransient<RandomService>();
var app = builder.Build();
```

```
var services = builder.Services;

app.MapGet("/", () => "Hello World!");
app.UseRouting();

app.MapGet("/services/list", async context =>
{
    var services = builder.Services;
    var middleware = new ServicesListMiddleware(null, services);
    await middleware.InvokeAsync(context);
});

app.MapGet("/services/timer", async context =>
{
    var timerService = app.Services.GetService<TimerService>();
    await context.Response.WriteAsync(
          $"Date and Time: {timerService?.GetCurrentDateTime()}");
});

app.MapGet("/services/random", async context =>
{
    //var timerService = app.Services.GetService<TimerService>();
    var randomNumber = app.Services.GetService<RandomService>();
    var num = randomNumber?.GetRandomNumber();
    await context.Response.WriteAsync(
          $"Random number: 1 - {num} 2 - {num}");
});

app.Run();
```

Лістинг RandomService:

```
namespace AspDotNetLab2
{
    public class RandomService
    {
        private readonly Random _random;

        public RandomService()
        {
            _random = new Random();
        }

        public int GetRandomNumber()
        {
            return _random.Next();
        }
    }
}
```

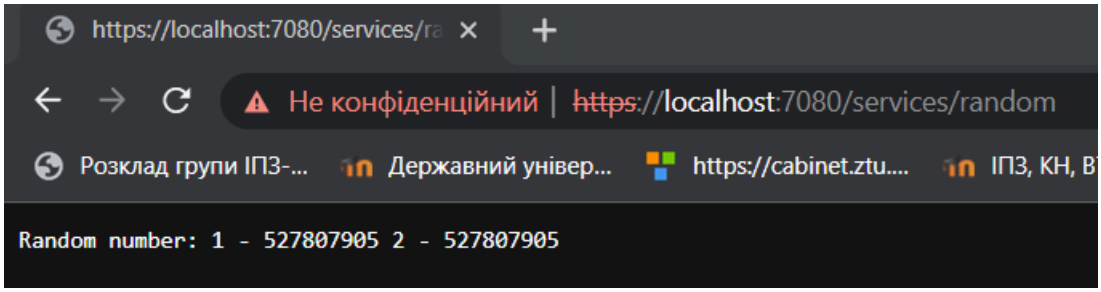


Рис.2.2. результат виконання програми

2.3. Тип “Singleton”, назва “GeneralCounterService” - веде підрахунок кількості звернень до кожної URL-адреси. Оскільки сервіс матиме тип “Singleton”,

то підрахунок кількості буде вестися для усіх запитів. Результати вивести за адресою “/services/general-counter”

Лістинг програми:

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Http;
using Microsoft.Extensions.DependencyInjection;
using AspDotNetLab2;
using System.Text;

var builder = WebApplication.CreateBuilder(args);
builder.Services.AddTransient<TimerService>();
builder.Services.AddTransient<RandomService>();
builder.Services.AddTransient<GeneralCounterService>();
var app = builder.Build();
var services = builder.Services;


app.MapGet("/", () => "Hello World!");
app.UseMiddleware<GeneralCounterMiddleware>();

app.UseRouting();


app.MapGet("/services/list", async context =>
{
    var services = builder.Services;
    var middleware = new ServicesListMiddleware(null, services);
    await middleware.InvokeAsync(context);
});

app.MapGet("/services/timer", async context =>
{
    var timerService = app.Services.GetService<TimerService>();
    await context.Response.WriteAsync(
           $"Date and Time: {timerService?.GetCurrentDateTime()}");
});

app.MapGet("/services/random", async context =>
{
    //var timerService = app.Services.GetService<TimerService>();
    var randomNumber = app.Services.GetService<RandomService>();
    var num = randomNumber?.GetRandomNumber();
    await context.Response.WriteAsync(
           $"Random number: 1 - {num} 2 - {num}");
});

app.Map("/services/general-counter", appBuilder =>
{
    appBuilder.Run(async context =>
    {
        var counterService =
context.RequestServices.GetRequiredService<GeneralCounterService>();
        var counters = counterService.GetCounters();

        await context.Response.WriteAsync("Counters:\n");
        foreach (var counter in counters)
        {
            await context.Response.WriteAsync($"{counter.Key}: {counter.Value}\n");
        }
    });
```

```
});

app.Run();
```

Лістинг GeneralCounterService**:**

```
using System.Collections.Concurrent;
using System.Diagnostics.Metrics;

namespace AspDotNetLab2
{
    public class GeneralCounterService
    {

        private readonly ConcurrentDictionary<string, int> _counters;

        public GeneralCounterService()
        {
            _counters = new ConcurrentDictionary<string, int>();
        }

        public void IncrementCounter(string url)
        {
            _counters.AddOrUpdate(url, 1, (_, count) => count + 1);
        }

        public IReadOnlyDictionary<string, int> GetCounters()
        {
            return _counters;
        }
    }
}
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;

namespace AspDotNetLab2
{
    public class GeneralCounterMiddleware
    {
        private readonly RequestDelegate _next;
        private readonly GeneralCounterService _counterService;

        public GeneralCounterMiddleware(RequestDelegate next, GeneralCounterService
counterService)
        {
            _next = next;
            _counterService = counterService;
        }

        public async Task InvokeAsync(HttpContext context)
        {
            // Збільшити лічильник передаваної URL-адреси
            _counterService.IncrementCounter(context.Request.Path);

            // Продовжити обробку запиту
            await _next(context);
        }
    }
}
```

Рис.2.3. результат виконання програми

Посилання на репозиторій GitHub: https://github.com/Vika-Myronova/ASPDotNetLab2

Висновок: в ході виконання лабораторної роботи я ознайомилася зі структурою ASP.NET Core 7 додатків, навчилася створювати найпростіші та використовувати наявні компоненти middleware у ASP.NET Core 7.