LABORATORY ASSIGNMENT №1

Introduction to ASP.NET Core

Goal: get acquainted with the basic principles of .NET, нlearn how to configure the development environment and install the necessary components, acquire skills in creating solutions and projects of various types, acquire skills in processing requests using middleware.

Workflow:

Task 1. Installation of integrated development environment (IDE) and necessary workloads

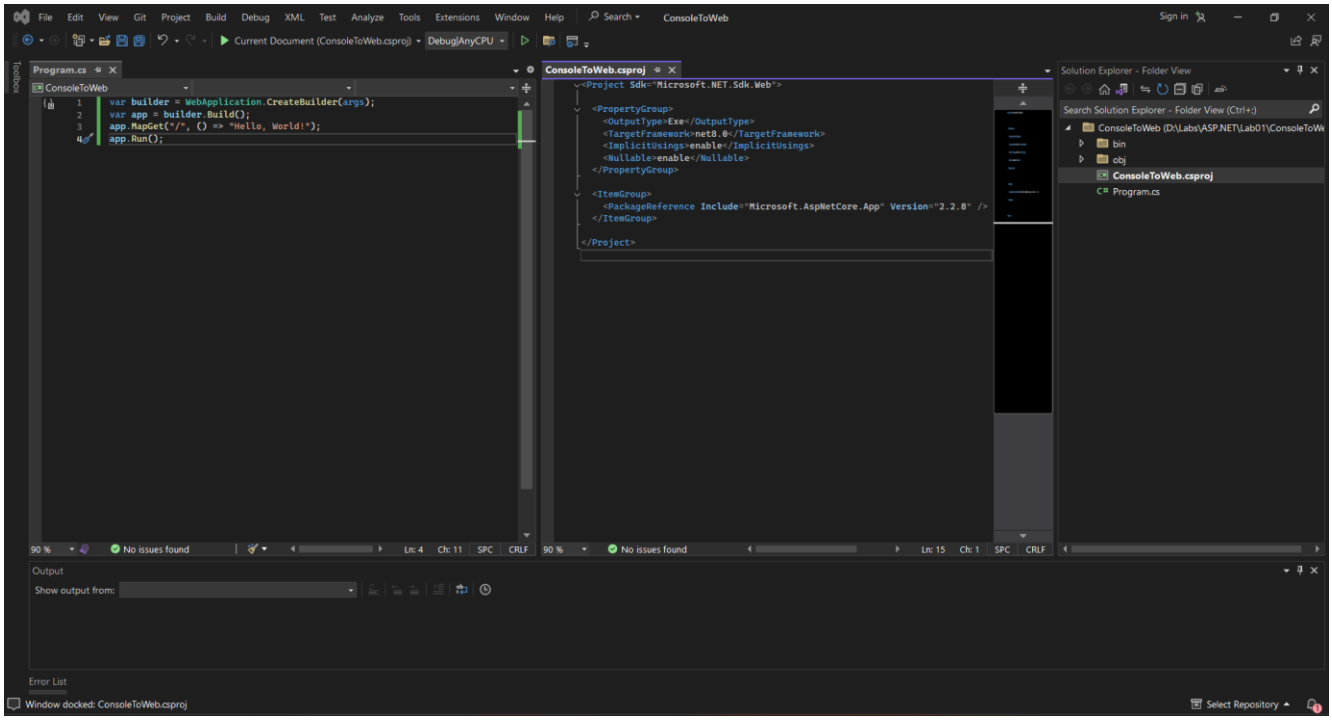
I've already had all installed, so I will skip this step

Task 2. Project creation

Part 1.

1. Create a project for a console application called ConsoleToWeb using the dotnet CLI
2. Convert the created console application into a web application
3. Describe the steps taken in the report

First I created console application by entering: *dotnet new console -n ConsoleToWeb* in console, then converted it into web application by editing SDK to *Microsoft.NET.Sdk.Web* in ConsoleToWeb.csproj file. After that I installed necessary dependencies by running: *dotnet add package Microsoft.AspNetCore.App*.



Part 2.

1. Create an ASP.NET WebAPI project without authorization named WebFromCli using dotnet CLI
2. Implement a GET endpoint “/who” that will return your first and last name
3. Implement a GET endpoint “/time” that will return the current time on the server
4. Provide a listing of implemented handlers in the report

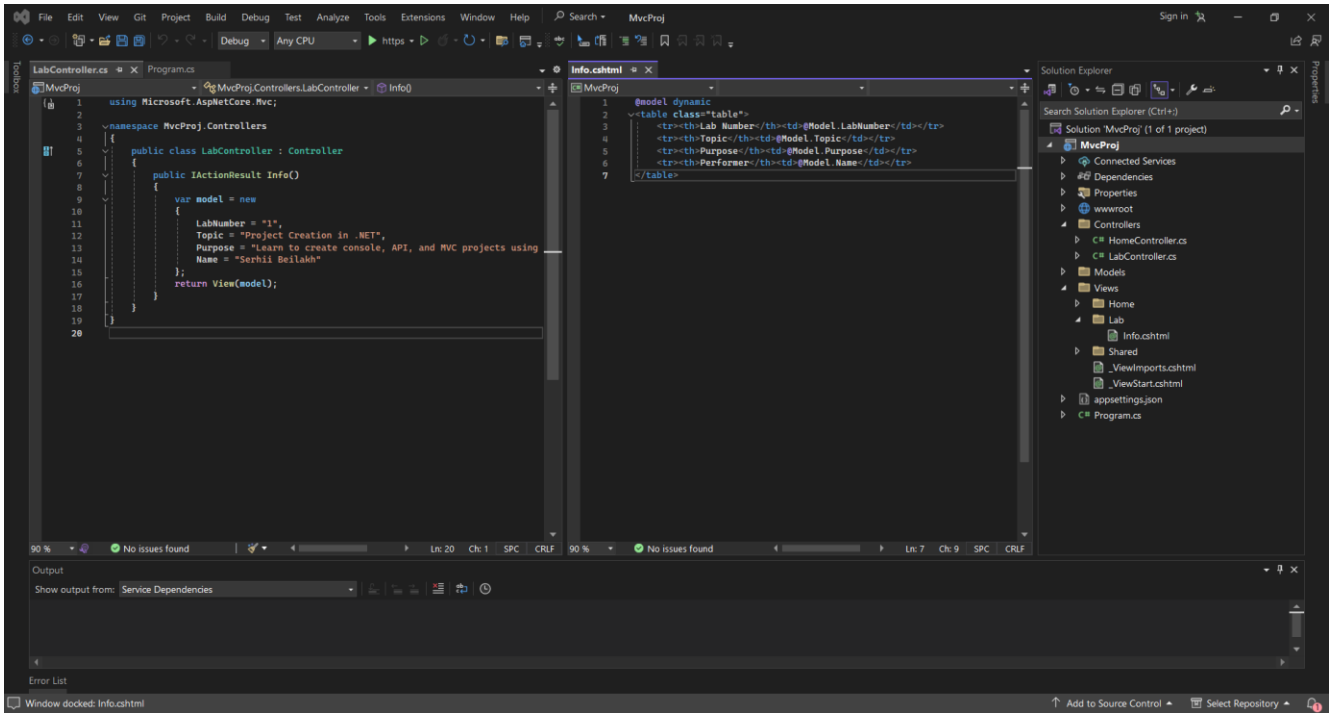
I created project by entering this command in console: *dotnet new webapi -n WebFromCli --auth None*.

Listing of implemented handlers:

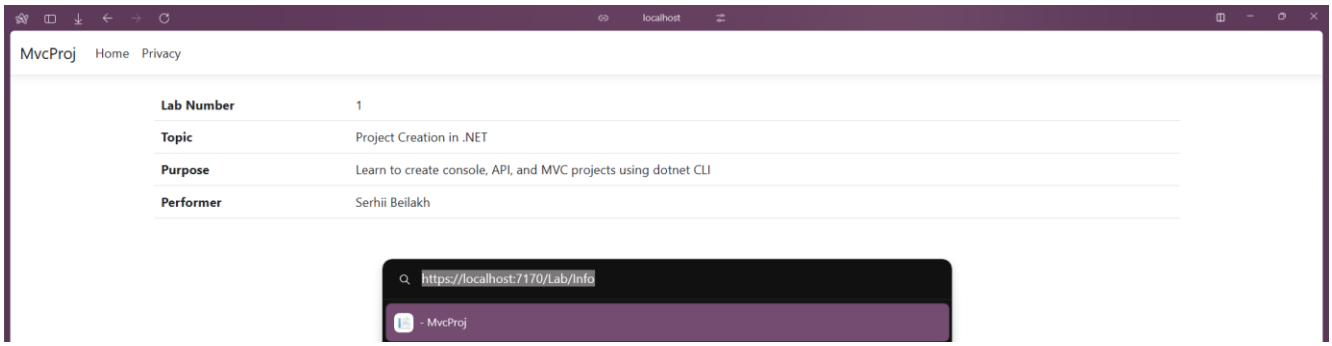
```
var builder = WebApplication.CreateBuilder(args);
var app = builder.Build();

app.MapGet("/who", () => "Serhii Beilakh");
app.MapGet("/time", () => DateTime.UtcNow.ToString("yyyy-MM-dd HH:mm:ss"));

app.Run();
```

Part 3.

1. Create an ASP.NET MVC project in any convenient way
2. Implement a controller named LabController
3. In the created controller, implement the /info handler, which will return a View with data about the laboratory work number, topic, purpose, and the name and surname of the performer in a table.
4. Data for table rendering should be transferred from the controller



Previous photo is controller and view, here is the result:



Task 3. Middlewares

1. Learn about the concept of middleware in ASP.NET

2. Create an ASP.NET WebAPI project named MiddlewareSandbox

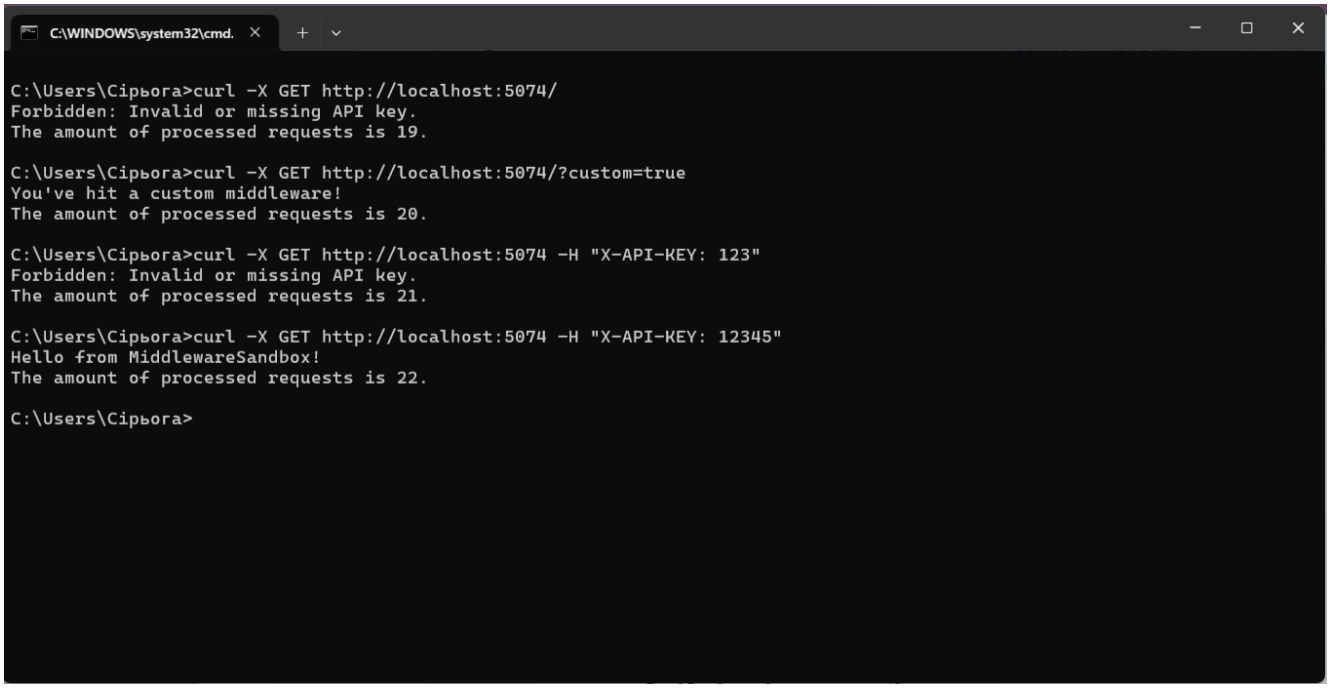
3. Implement the following middlewares

a. Implement a middleware that counts the number of requests to the server and returns this number in the response, for example: The amount of processed requests is X.

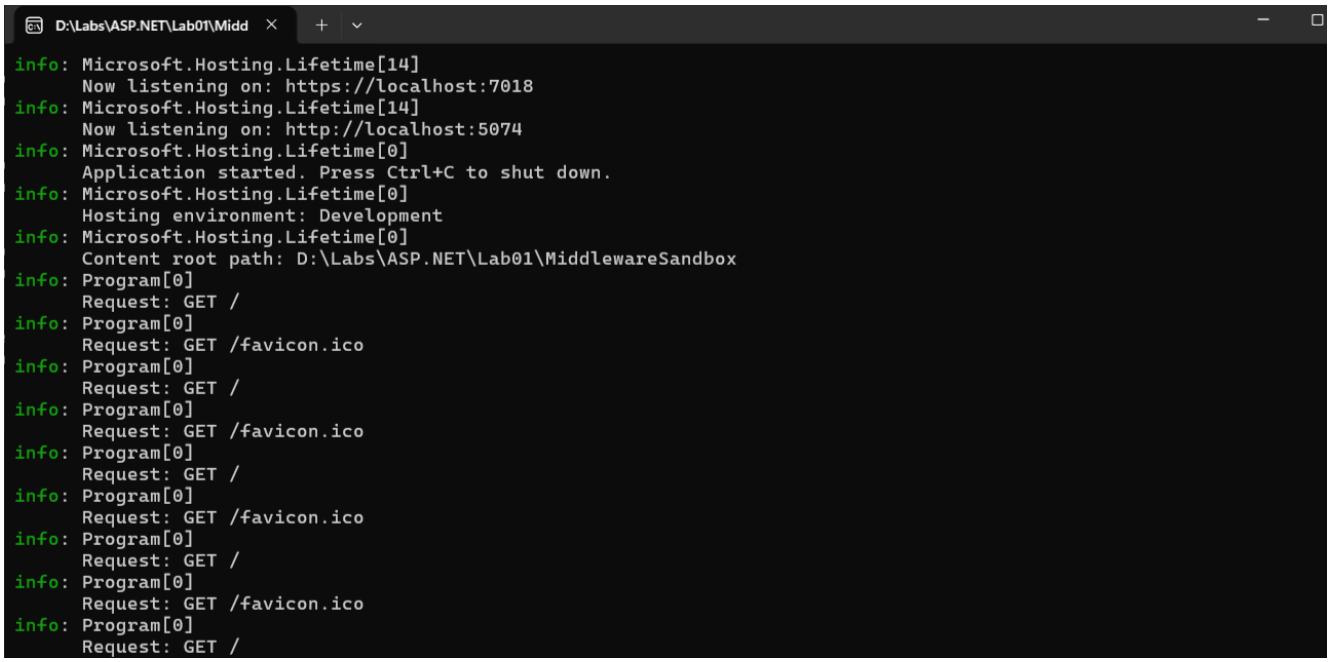
b. Implement a middleware that parses the query string parameters. If the query string contains the parameter “custom”, then return “You’ve hit a custom middleware!”, otherwise skip the request. An example of a request URL is shown in Figure 3.1.

c. Implement a middleware that logs all requests to the console. The logs should show the request method (GET, POST, etc.) and its path (/user, /product, etc.). You can use Postman or other similar utility to test POST and other request methods.

d. Implement middleware that checks for the presence of an API key in the request headers. To do this, check whether the request contains the X-API-KEY header. If the key is incorrect or missing, return 403 Forbidden, do not pass the request on for further execution. If the key in the request header matches the one specified on the server, pass the request on. You can use Postman or similar utilities to add the corresponding request.



Testing middleware in console.



Logger

Conclusion: I got acquainted with the basic principles of .NET, learned how to configure the development environment and install the necessary components, acquired skills in creating solutions and projects of various types, and skills in processing requests using middleware.