Web API bảo mật với IdentityServer4: IdentityServer4 với .Net Core Phần III

Neel .Net lõi , .Net Lõi 2.1 , .Net Lõi an ninh , .API , .ASP Net lõi , .ASP Net Lõi 2.0 , .Asp Net Lõi 2.1 , .asp.netcore2.0 , .Authentication trong Net Lõi 2.0 , .IdentityServer4 , .IdentityServer4 thiết lập , .IdentityServer4 với .Net Core , .Bảo mật trong .Net Core , .Visual Studio 2017 Ngày 8 tháng 3 năm 2018 5 phút

Lưu ý - Bạn có thể tìm thấy mã nguồn của ứng dụng mẫu của tôi tại đây (https://github.com/NeelBhatt/TeamX.Security.AuthenticationServer-IdentityServerv4-) .

Bạn có thể tìm thấy tất cả . Net cốt lõi bài viết ở đây (https://neelbhatt.com/category/net-core/) .

Trong bài trước của tôi trên **IdentityServer4** , tôi đã giải thích cách thiết lập một **máy chủ Auth** và cũng tạo ra một **máy khách** . Bạn có thể tìm thấy bài đăng ở đây (https://neelbhatt.com/2018/03/04/step-by-step-setup-for-the-auth-server-and-the-client-identityserver4-with-net-core-part-ii/) .

Tôi sẽ yêu cầu bạn đi qua bài đăng trước đó (https://neelbhatt.com/2018/03/04/step-by-step-setup-for-the-auth-server-and-the-client-identityserver4-with-net-core-part-ii/) trước khi đọc bài đăng này.

Trong bài này, chúng ta hãy bảo mật một **API** bằng **IdentityServer4** .

Vì vậy, trong bài viết này:

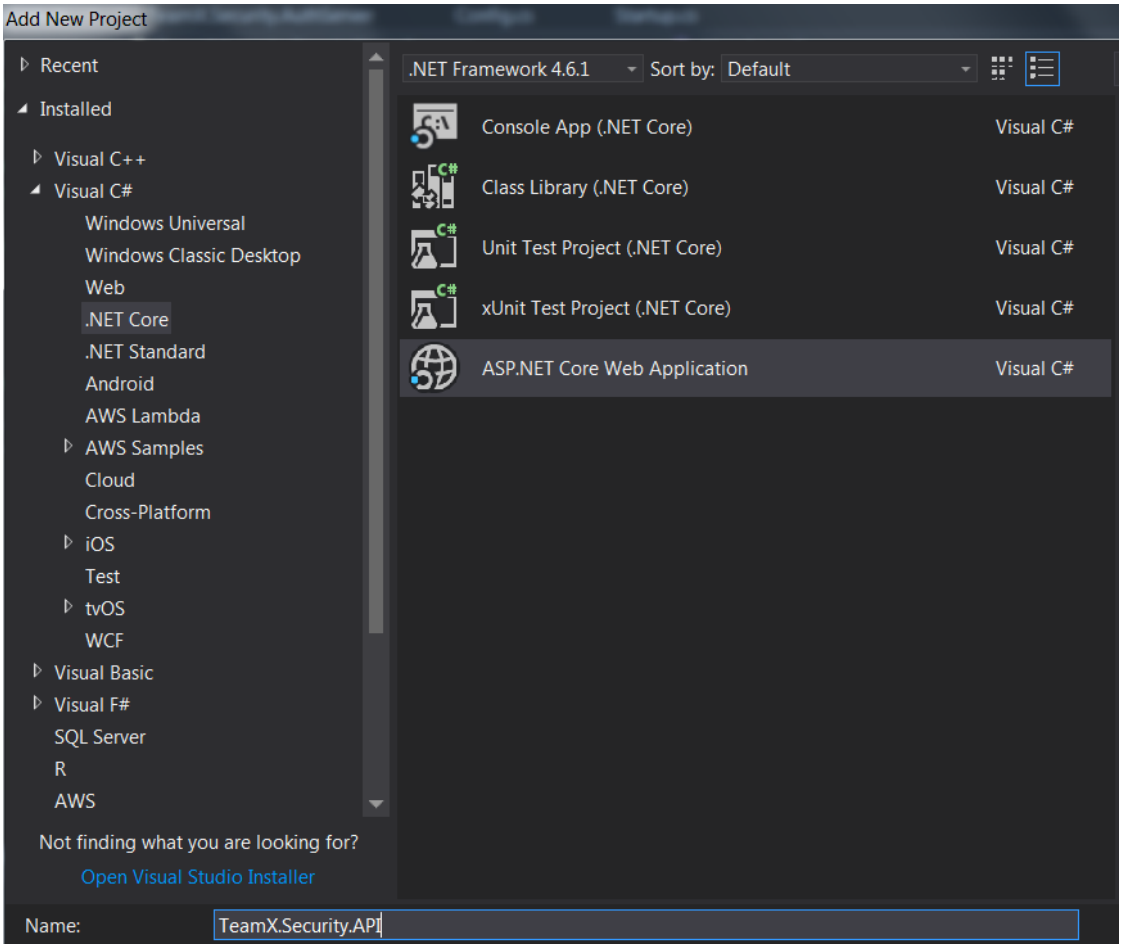
- Chúng tôi sẽ tạo một **API** và sẽ bảo mật bộ điều khiển API với thuộc tính **Authorize**
- Chúng tôi cũng sẽ thêm logic để **tiêu thụ mã thông báo** từ **máy chủ Auth** trong API
- Sau đó, chúng tôi sẽ sửa đổi **máy khách** (mà chúng tôi đã tạo trong bài trước) để gọi **API bằng cách sử dụng mã thông báo** được tạo bởi máy chủ Auth

- Cuối cùng, chúng tôi sẽ nhận được **xác nhận quyền sở hữu** từ API vào bảng điều khiển của khách hàng nếu mã thông báo được xác thực. Do đó, API được truy cập nhưng **theo cách bảo mật**

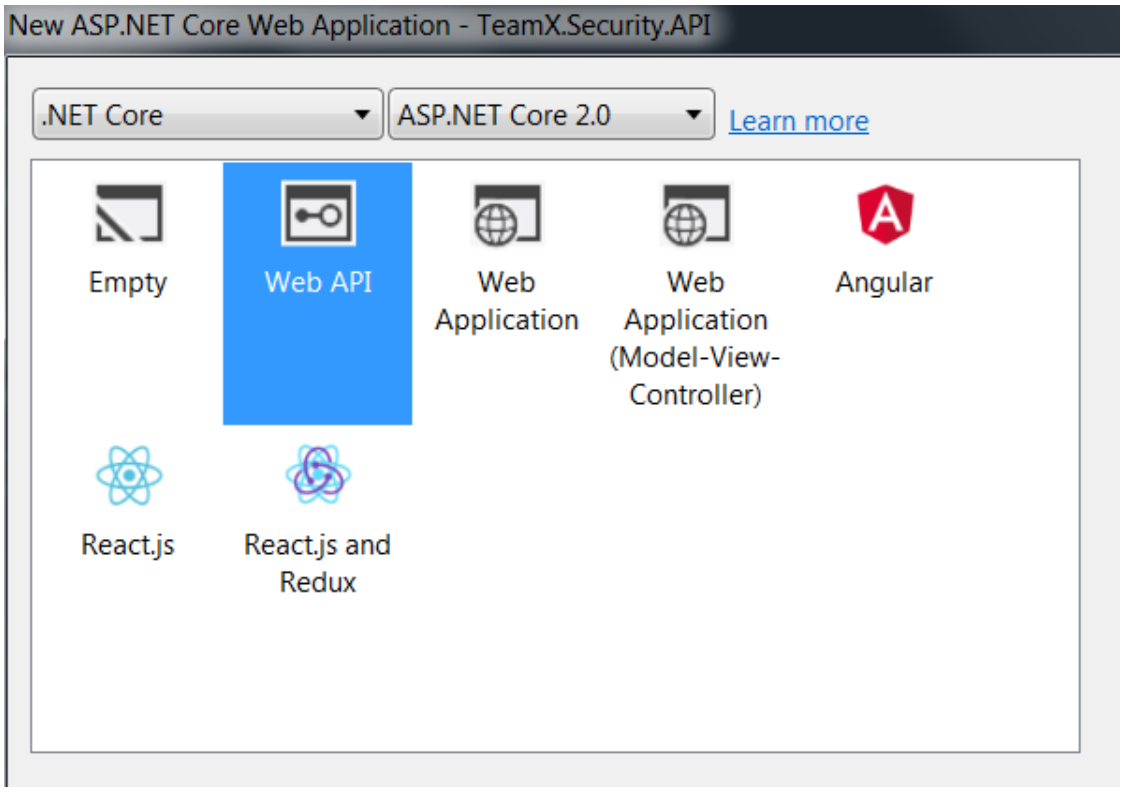
Thú vị đúng không? Chúng ta hãy bắt đầu.

API

Nhấp chuột phải vào giải pháp -> Tạo dự án mới -> Chọn **ứng dụng Web lõi** :



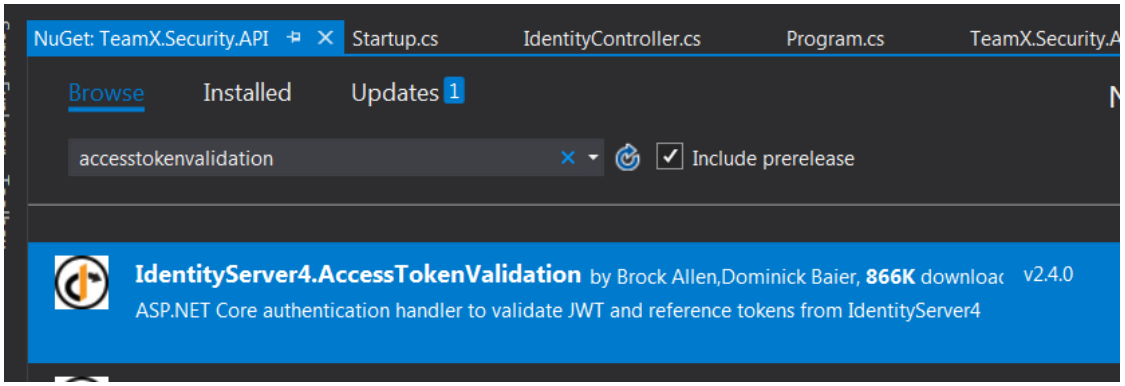
Nhấp vào Ok và trong cửa sổ tiếp theo, chọn dự án Web API như hình dưới đây:



Vì chúng tôi muốn bảo mật các **API của mình bằng các mã thông báo** , API của chúng tôi cần có khả năng **tiêu thụ mã thông báo** từ **máy chủ Auth** và **hạn chế người dùng** tương ứng. Do đó chúng tôi sẽ cần các chức năng mà sẽ làm quá trình này cho chúng tôi.

Đối với điều này, chúng ta cần phải cài đặt **IdentityServer.AccessTokenValidation** trong ứng dụng. Đây là **trình xử lý xác thực** để xác nhận **mã thông báo JWT** và r **eference** từ IdentityServer4

Mở Nuget và tìm kiếm bằng **IdentityServer.AccessTokenValidation** -> nhấp vào **cài đặt** :



Khi gói đã được cài đặt, chúng ta sẽ tạo một **bộ điều khiển** mà chúng ta sẽ bảo mật bằng cách thêm thuộc tính **Authorize** . Chúng tôi sẽ chỉ cho phép những người dùng đó truy cập vào **API** gửi **mã thông báo truy cập** chính xác .

Chúng ta hãy tạo một **IdentityController** , nó sẽ trả về các **yêu cầu** của người dùng khi được gọi.

Thêm mã bên dưới vào IdentityController:

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

namespace TeamX.Security.API.Controllers
{
[Route("[controller]")]
[Authorize]
public class IdentityController : Controller
{
[HttpGet]
public IActionResult Get()
{
return new JsonResult(from c in User.Claims select new { c.Type, c.Value });
}
}
}
```

Ở đây chúng tôi có:

- Đã thêm một thuộc tính **Authorize** sẽ được sử dụng để hạn chế người dùng truy cập vào bộ điều khiển
- Đã thêm **phương thức Nhận** sẽ được sử dụng để trả **lời xác nhận quyền sở hữu của người dùng**

Bước tiếp theo là **thêm xác thực** trong lớp **Startup.cs** của **dự án API** . ở đây chúng tôi đang thêm các chi tiết cần thiết cho Máy chủ Identity.

Thêm mã bên dưới vào phương thức **ConfigureServices** :

```
public void ConfigureServices(IServiceCollection services)
{

services.AddMvcCore()
.AddAuthorization()
.AddJsonFormatters();

services.AddAuthentication("Bearer")
.AddIdentityServerAuthentication(options =>
{
options.Authority = "http://localhost:5000 (http://localhost:5000)";
options.RequireHttpsMetadata = false;

options.ApiName = "api1";
});

}
```

Ở đây chúng tôi có:

- Thêm **xác thực** bằng cách sử dụng **AddAuthentication** sẽ thêm các **dịch vụ xác thực** vào **DI** và cấu hình **Bearer** làm **lược đồ mặc định**
- **AddIdentityServerAuthentication** thêm **trình xử lý xác thực** mã thông báo truy cập **IdentityServer** vào DI để sử dụng bởi các **dịch vụ xác thực**

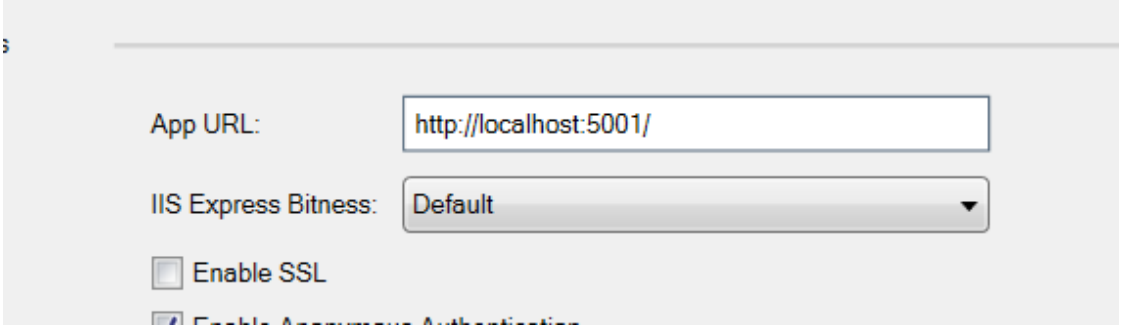
Chúng tôi đã chỉ định chi tiết máy chủ Auth.

Bây giờ chúng ta hãy thêm **UseAuthentication** vào phương thức **Configure** :

```
1  public void Configure(IApplicationBuilder app)
2  {
3  app.UseAuthentication();
4
5  app.UseMvc();
6  }
```

UseAuthentication thêm **phần mềm trung gian xác thực** vào đường dẫn để **xác thực** sẽ được thực hiện **tự động** trên **mọi cuộc gọi vào máy chủ.**

Bước cuối cùng là thiết lập cổng của dự án **API** thành **5001** như hình dưới đây:



Đó là nó. Đã đến lúc kiểm tra mọi thứ cùng nhau.

Trước khi chạy dự án của chúng tôi, chúng tôi cần thực hiện các thay đổi trong **ứng dụng khách** mà chúng tôi đã tạo trong bài đăng trước. Chúng tôi cần thêm logic để **gọi API của chúng tôi** bằng cách **gửi mã thông báo** sẽ được tạo từ **máy chủ Auth** .

Thêm mã bên dưới vào ứng dụng khách:

```
// call api
var client = new HttpClient();
client.SetBearerToken(tokenResponse.AccessToken);

var response = await client.GetAsync("http://localhost:5001/identity (http:/
if (!response.IsSuccessStatusCode)
{
Console.WriteLine(response.StatusCode);
}
else
{
var content = await response.Content.ReadAsStringAsync();
Console.WriteLine(JArray.Parse(content));
}
```

Ở đây chúng tôi là:

- Tạo **HTTPClient** và thiết lập **mã thông báo mang** mà chúng tôi nhận được từ **máy chủ Auth**
- Chúng tôi sẽ gọi **API** được lưu trữ trên cổng **5001** và sẽ gọi **bộ điều khiển nhận dạng**
- API sẽ **khớp với mã thông báo** và nếu người dùng được **ủy quyền** , API sẽ cho phép yêu cầu nhập vào phương thức Nhận của bộ điều khiển
- Các **tuyên bố của người dùng** sẽ được trả lại cho khách hàng

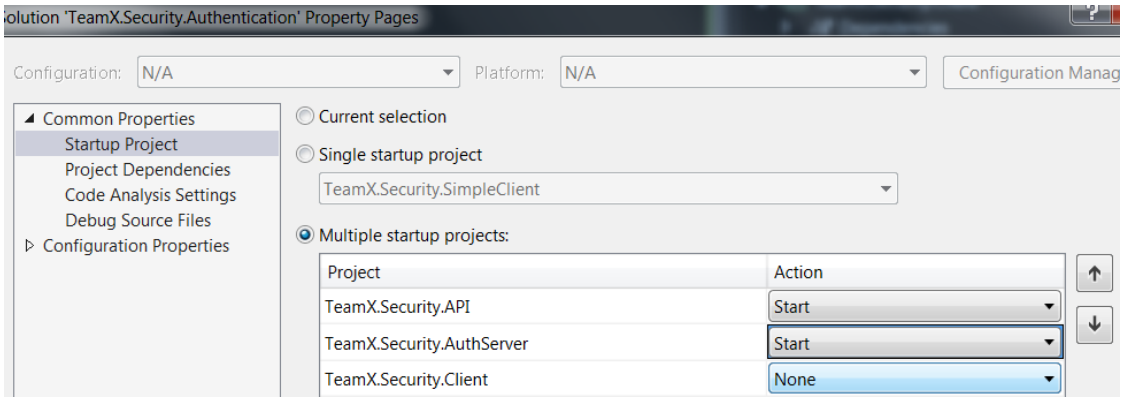
Mã toàn bộ khách hàng sẽ trông giống như dưới đây:

```
using IdentityModel.Client;
using Newtonsoft.Json.Linq;
using System;
using System.Net.Http;
using System.Threading.Tasks;

namespace TeamX.Security.Client
{
class Program
{
public static void Main(string[] args) => MainAsync().GetAwaiter().GetResult

private static async Task MainAsync()
{
// discover endpoints from metadata
var disco = await DiscoveryClient.GetAsync("http://localhost:5000 (http://lo
if (disco.IsError)
{
Console.WriteLine(disco.Error);
return;
}

// request token
var tokenClient = new TokenClient(disco.TokenEndpoint, "client", "secret");
var tokenResponse = await tokenClient.RequestClientCredentialsAsync("api1");

if (tokenResponse.IsError)
{
```

```
Console.WriteLine(tokenResponse.Error);
return;
}

Console.WriteLine(tokenResponse.Json);
Console.WriteLine("\n\n");

// call api
var client = new HttpClient();
client.SetBearerToken(tokenResponse.AccessToken);

var response = await client.GetAsync("http://localhost:5001/identity (http:/
if (!response.IsSuccessStatusCode)
{
Console.WriteLine(response.StatusCode);
}
else
{
var content = await response.Content.ReadAsStringAsync();
Console.WriteLine(JArray.Parse(content));
}
}
}
}
```

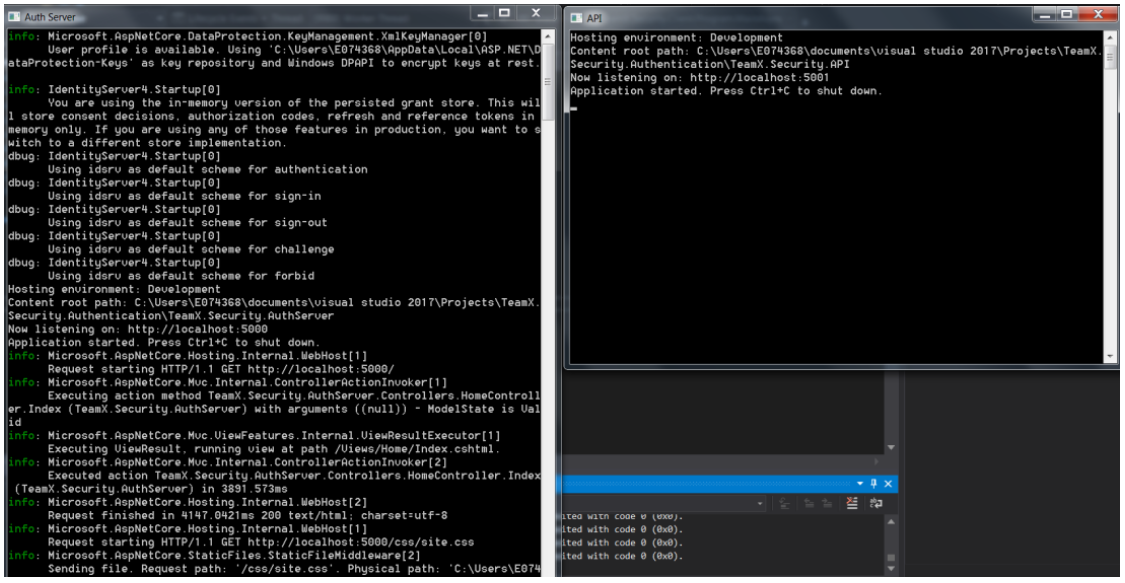
Chạy và kiểm tra ứng dụng

Vì chúng tôi muốn **bắt đầu nhiều dự án** cùng nhau, chúng tôi cần đặt nhiều dự án khởi động. Để làm điều đó, kích chuột phải vào **giải pháp** và mở **Properties** -> Set như hình dưới đây:



Sau khi mọi thứ được thiết lập, chỉ cần **bắt đầu** các **ứng dụng** .

Cả hai ứng dụng (**Auth Server** và **API**) sẽ được bắt đầu và giao diện điều khiển sẽ xuất hiện:

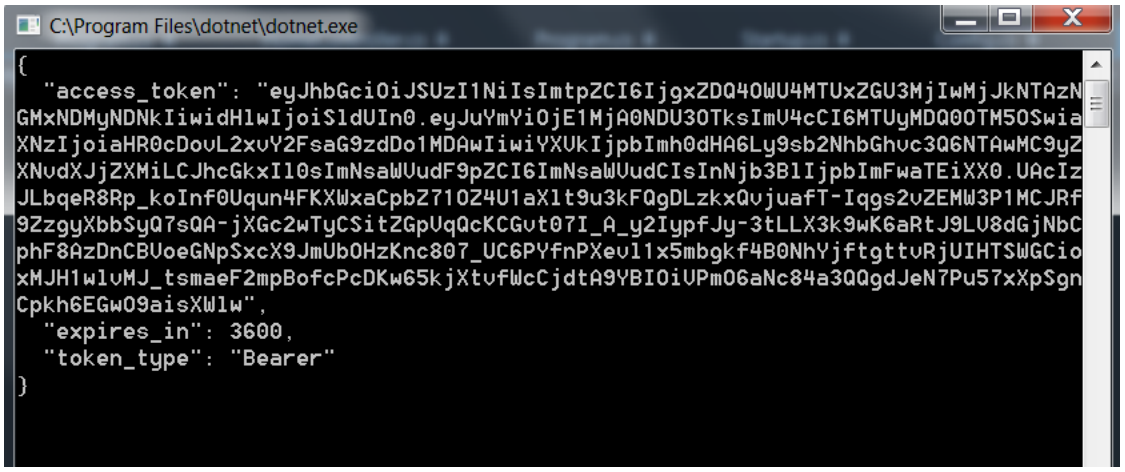


Như bạn có thể thấy ở trên, có một **đăng nhập** tốt đẹp xảy ra trong giao diện điều khiển. Trên đầu trang, có một **gợi ý** như sau:

> *Bạn đang sử dụng phiên bản trong bộ nhớ của cửa hàng cấp quyền liên tục. Thao tác này sẽ lưu trữ các quyết định chấp thuận, mã ủy quyền, làm mới và mã thông báo tham chiếu trong bộ nhớ. Nếu bạn đang sử dụng bất kỳ tính năng nào trong số các tính năng đó trong sản xuất, bạn muốn làm phù thủy với một triển khai cửa hàng khác.*

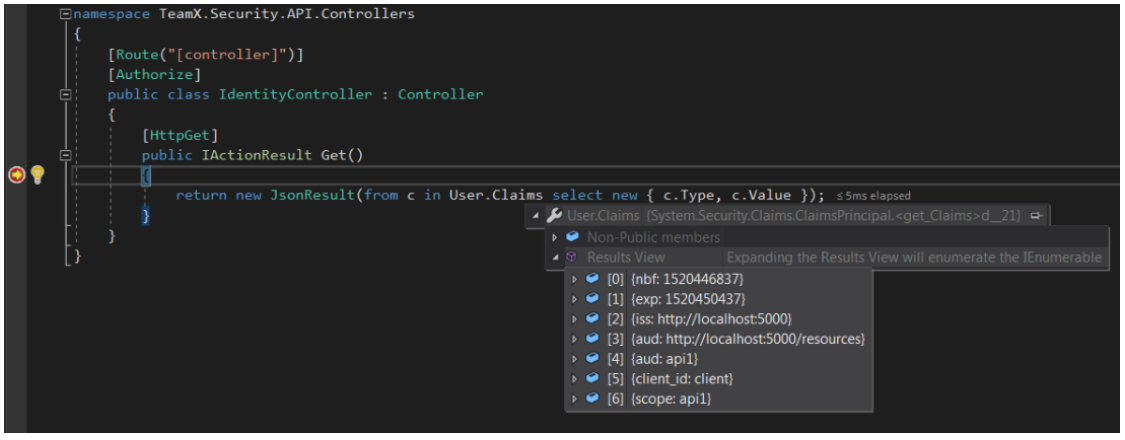
Khi cả hai dự án đang chạy, chúng ta **hãy bắt đầu ứng dụng khách** của chúng tôi **.**

Khi chúng tôi chạy **ứng dụng khách** , trước tiên máy khách sẽ **yêu cầu mã thông báo** từ **máy chủ Auth** và **mã thông báo truy cập** sẽ được **máy chủ Auth** gửi đi như sau:



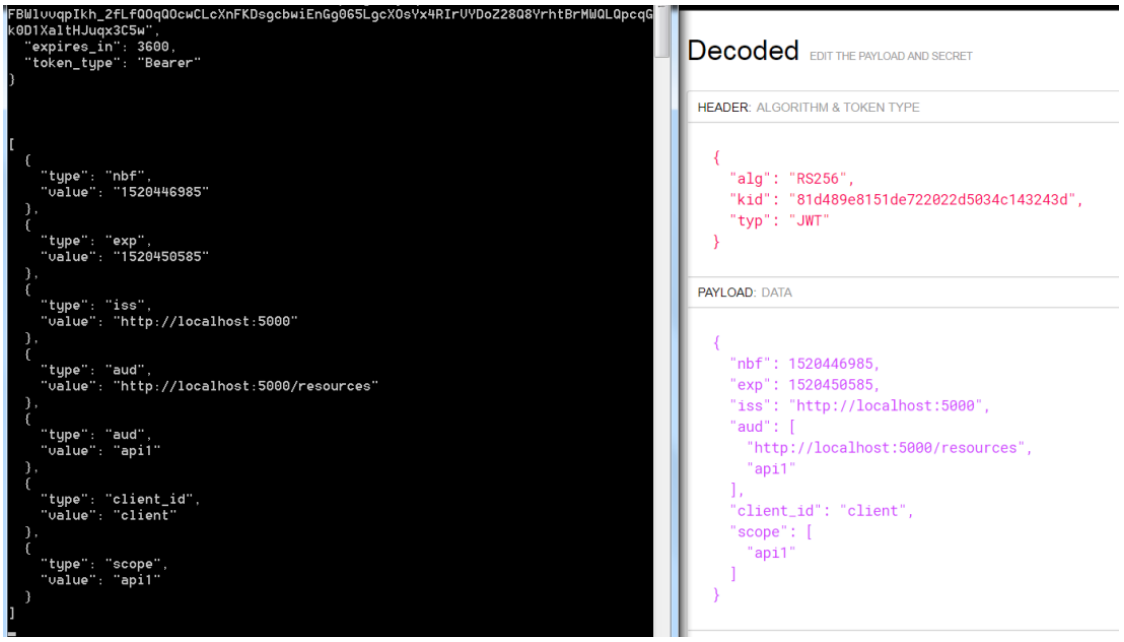
Mã thông báo này sẽ được gửi đến **API** và nếu **mã thông báo khớp với** thì phương thức Nhận của API sẽ được truy cập.

Chúng tôi có thể xem các xác nhận quyền sở hữu của người dùng như được hiển thị bên dưới:



Do đó, API sẽ trả về xác nhận quyền sở hữu của người dùng:

Bạn có thể **xác nhận xác nhận quyền sở hữu** bằng cách truy cập trang web - https://jwt.io/ (https://jwt.io/) . Cung cấp mã thông báo truy cập trong hộp văn bản được **mã hóa** và nó sẽ trả về **chi tiết xác nhận quyền sở hữu** . Như bạn có thể thấy bên dưới, cả hai giá trị đều khớp nhau - có nghĩa là chúng tôi đã thiết lập chính xác:



Trong các bài báo sau, chúng ta sẽ sử dụng .Net Core Identity với IdentityServer4 và cũng sẽ tạo một máy khách MVC.

Lưu ý - Bạn có thể tìm thấy mã nguồn của ứng dụng mẫu của tôi tại đây (https://github.com/NeelBhatt/TeamX.Security.AuthenticationServer-IdentityServerv4-) .

Hy vọng nó giúp.

Nhà xuất bản Neel

Xem những bài viết của Neel

2 suy nghĩ về " An ninh Web API với IdentityServer4: IdentityServer4 với .Net Core Phần III "

Pingback: Dew Drop - ngày 12 tháng 3 năm 2018 (# 2681) - Morning Dew
Srini nói:
11 tháng 7 năm 2018 lúc 3:46 chiều
Có lẽ hướng dẫn tốt nhất về lõi .net 2 với máy chủ định danh 4.

 Đáp lại