

[github](#) - [Stack Overflow](#) - [LinkedIn](#) - [MVA](#) - a developer braindump on .Net, c#, Java, Javascript ...

**{"@id":"cedric-dumont.com"}**

## IdentityServer.v3, MembershipReboot, AngularJs, WebApi 2 and MVC : Mix It ! : Part 5

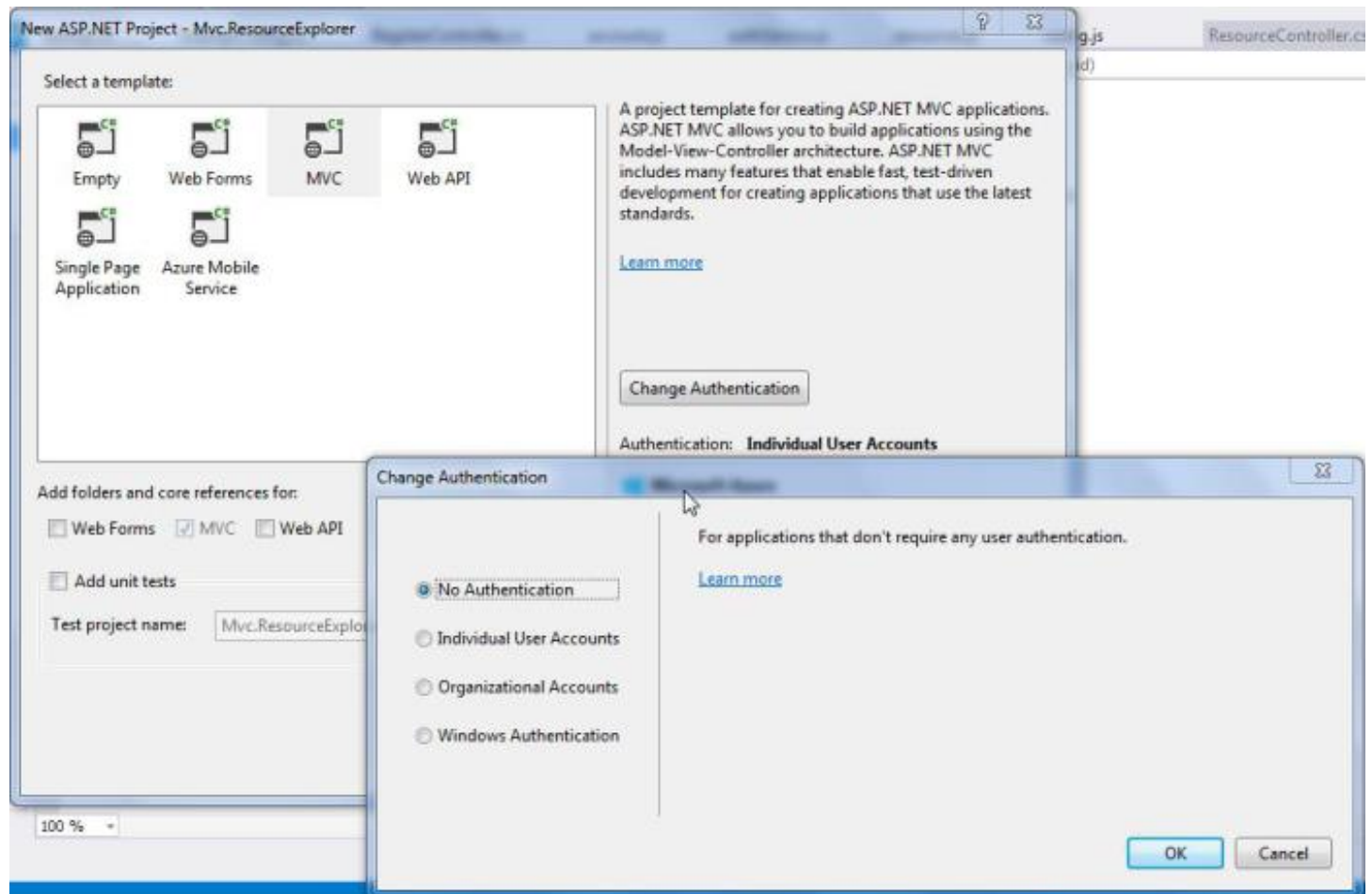
**Home (<http://cedric-dumont.com/tutorials/identityserver-v3-membershipreboot-angularjs-webapi-2-and-mvc-mix-it-introduction/>)**

In This part of the tutorial will deal with creating a client that will access our public api after we gave consent to it.

### **Step 1 : create the project**

In visual studio :

- create a new Web project
- select the MVC check box
- change authentication to : No Authentication



(<https://cedricdumont.files.wordpress.com/2014/12/5-1.jpg>)

## Step 2 : create the implicit client

In our Identity server, we need to create an implicit client. we do this by adding the following in the *Clients.cs* file (check the RedirectUris that must match our application URL)

```

1  new Client
2  {
3      ClientName = "Resource Explorer",
4      Enabled = true,
5
6      ClientId = "ResourceExplorer",
7      ClientSecret = "secret",
8      Flow = Flows.Implicit,
9
10     ClientUri = "http://www.resExplorer.com (http://www.resExplorer.co
11
12     RequireConsent = true,
13     AllowRememberConsent = true,
14
15     RedirectUris = new List<string>
16     {
17         "http://localhost:10071/ (http://localhost:10071/)",
18     },
19
20     PostLogoutRedirectUris = new List<string>
21     {
22         "http://localhost:10071/ (http://localhost:10071/)",
23     },

```

```

24
25     IdentityTokenLifetime = 360,
26     AccessTokenLifetime = 3600
27 }

```

### Step 3 : Install some nuget

```

1  install-package Microsoft.Owin.Security.OpenIdConnect
2  install-package Thinktecture.IdentityModel.Client // for convenience
3  install-package Microsoft.Owin.Host.SystemWeb // to start owin
4  Install-Package Microsoft.Owin.Security.Cookies

```

### Step 4 : add the Startup.cs

add the Startup.cs file and OpenIdAuthentication middleware.

```

1  app.UseOpenIdConnectAuthentication(
2      new OpenIdConnectAuthenticationOptions
3      {
4          Authority = "https://localhost:44305/identity/ (https://localhos
5
6          ClientId = "ResourceExplorer",
7          Scope = "openid profile publicApi",
8          ResponseType = "id_token token",
9          // this project URL must be configured in client
10         RedirectUri = "http://localhost:10071/ (http://localhost:10071/)
11         SignInAsAuthenticationType = "Cookies",
12         UseTokenLifetime = false,

```

### Step 5 : add a controller and a view

The controller has some actions protected with the [Authorize] attribute that will forward to our identity server if not authenticated.

```

1  [Authorize]
2  public async Task<ActionResult> Resource()
3  {
4      ViewBag.RetrievedResource = await GetResourceWithId(1);
5
6      return View("Resource");
7  }

```

The *GetResourceWithId()* method calls the public web api using a bearer token (your url might be different and you could of course config it somewhere else).

```

1  private async Task<String> GetResourceWithId(Int32 id)
2  {
3      var user = User as ClaimsPrincipal;
4      var token = user.FindFirst("access_token").Value;
5
6      var client = new HttpClient();
7      client.SetBearerToken(token);
8

```

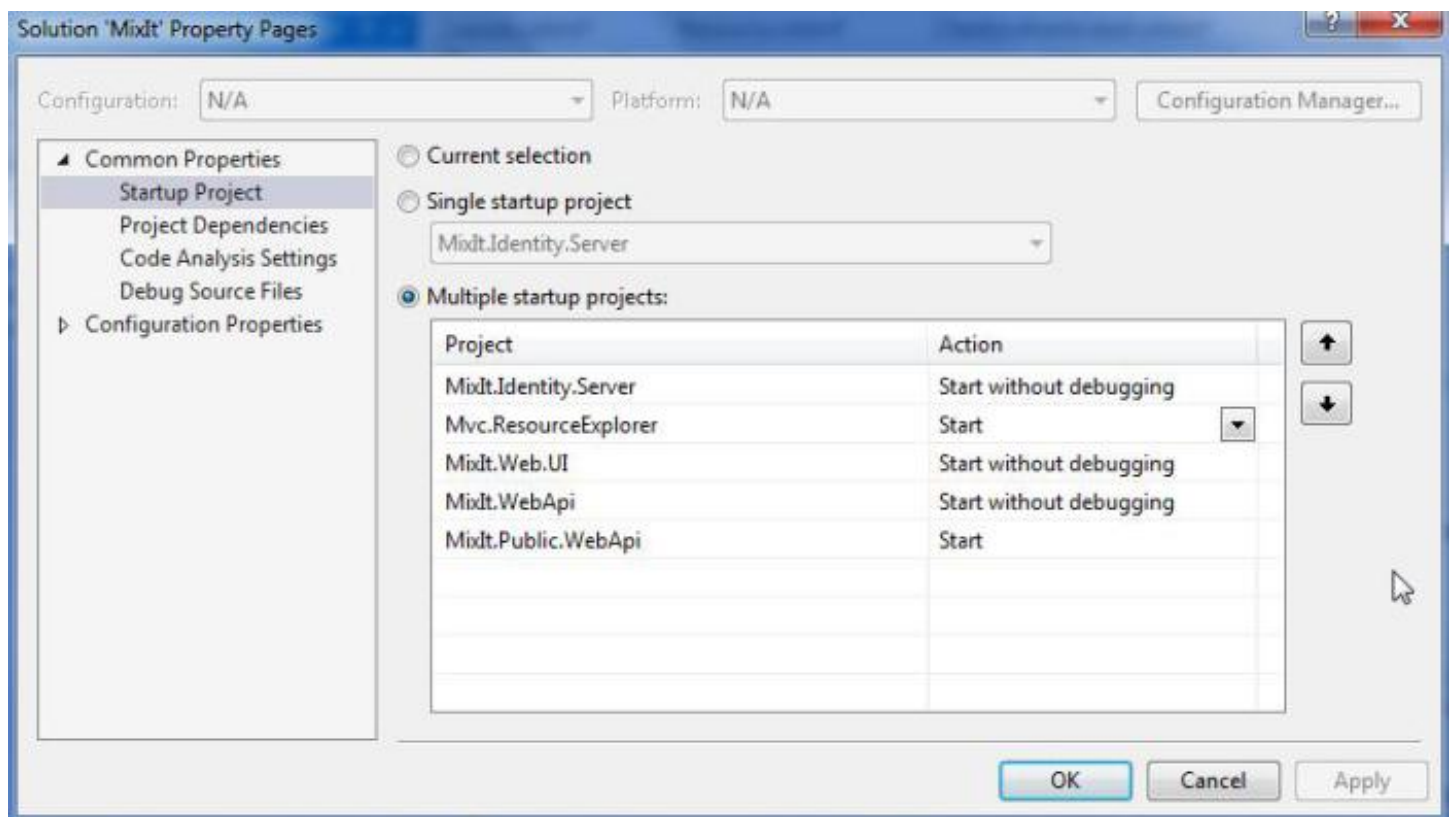
```

9      HttpResponseMessage response =
10          await client.GetAsync("http://localhost:14117/resource/ (http://
11
12      return await response.Content.ReadAsStringAsync();
13  }

```

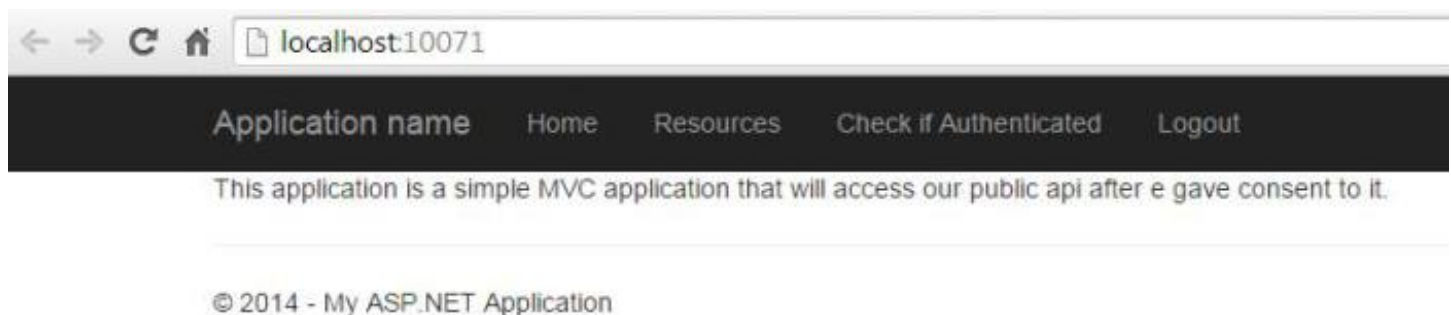
## Step 6 : Test It

You need to add the project in the solution Startup projects list (you could also have only the Identity.server and ResourceExplorer project running)



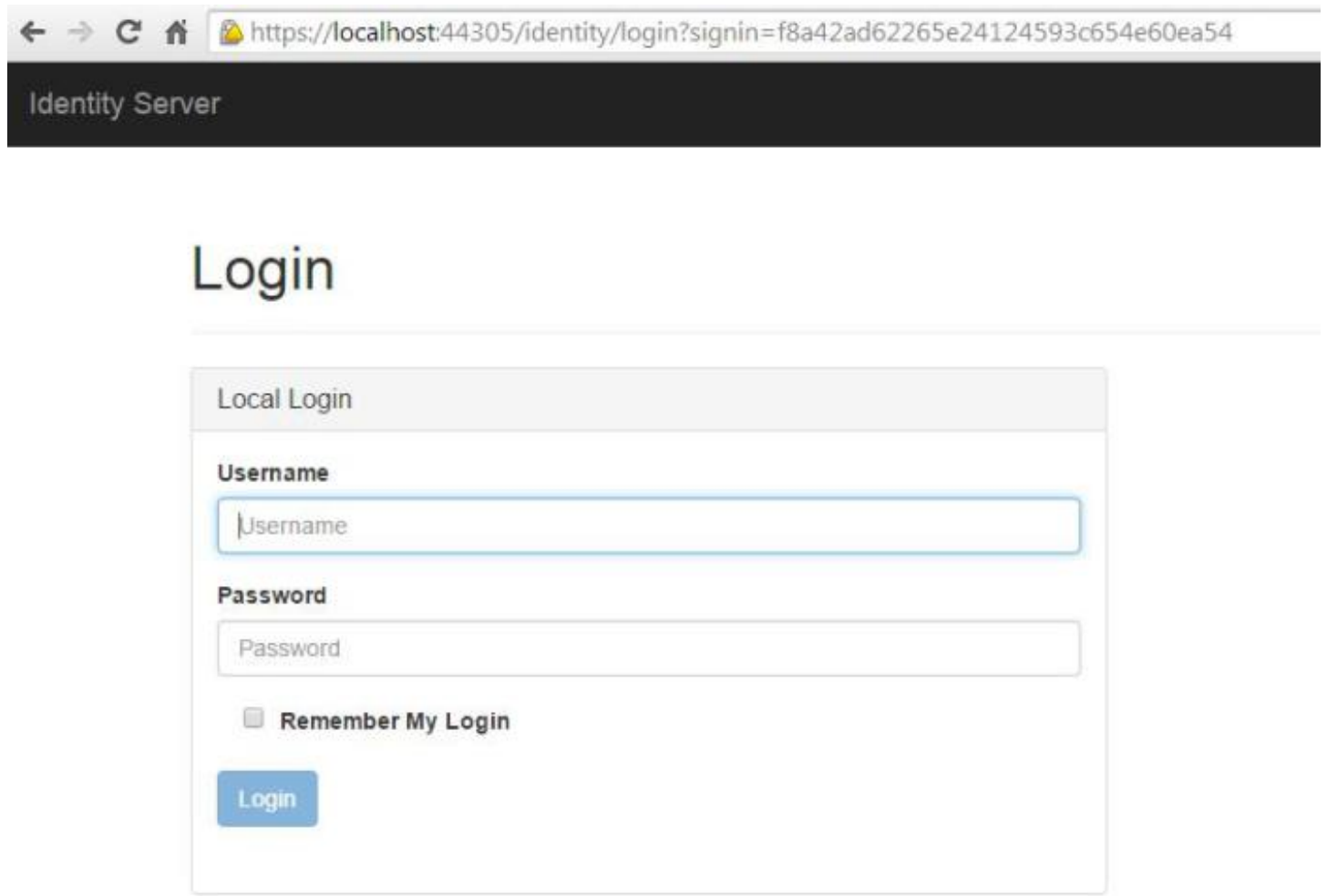
(<https://cedricdumont.files.wordpress.com/2014/12/5-2.jpg>)

Next start the solution and go to the MVC explorer page :



(<https://cedricdumont.files.wordpress.com/2014/12/5-3.jpg>)

When you click on the Resources Menu item, you will be forwarded to the Identity.server that will show up a login page.



← → ↻ 🏠 <https://localhost:44305/identity/login?signin=f8a42ad62265e24124593c654e60ea54>

Identity Server

## Login

Local Login

**Username**

**Password**


☐ Remember My Login

<https://cedricdumont.files.wordpress.com/2014/12/5-4.jpg>


After login, a consent screen will appear


## Resource Explorer is requesting your permission

Uncheck the permissions you do not wish to grant.

 Personal Information

☒ Your user identifier *(required)*

☒ User profile   
Your user profile information (first name, last name, etc.)


 Application Access

☒  
Access to our public API

☒ Remember My Decision

Yes, Allow

No, Do Not Allow

 Resource Explorer

<https://cedricdumont.files.wordpress.com/2014/12/5-5.jpg>

Allow the app to access your public api and you 'll be forwarded to the Resource page showing you link to get the Resources.

Application name

Home

Resources

Check if Authenticated

Logout

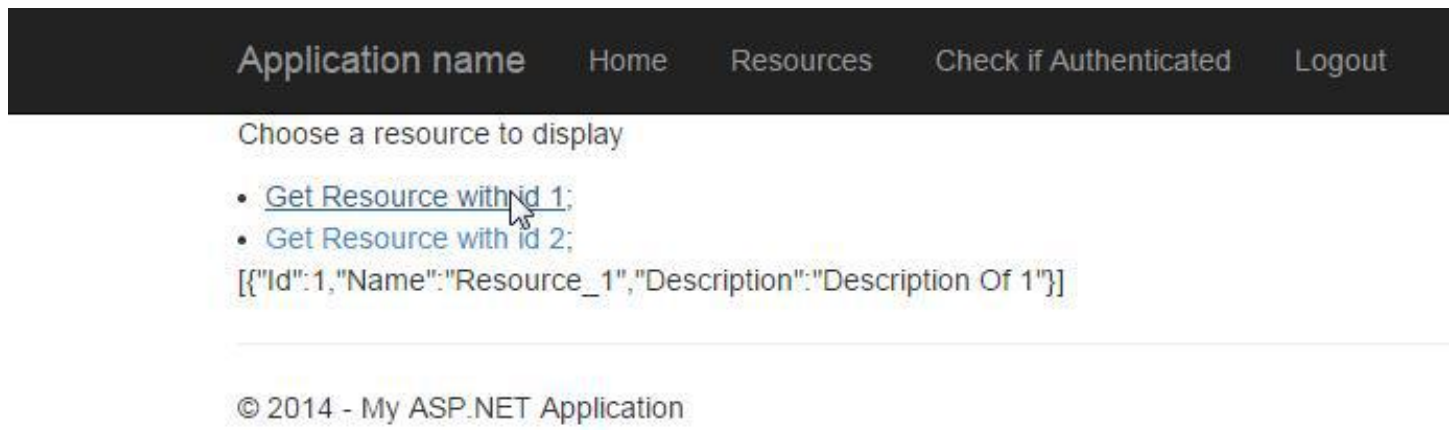
Choose a resource to display

- [Get Resource with id 1;](#)
- [Get Resource with id 2;](#)

© 2014 - My ASP.NET Application

<https://cedricdumont.files.wordpress.com/2014/12/5-6.jpg>

If you click a resource to get, it will display simply below without asking you to consent again



<https://cedricdumont.files.wordpress.com/2014/12/5-7.jpg>

That's It !!!

## One thought on “IdentityServer.v3, MembershipReboot, AngularJs, WebApi 2 and MVC : Mix It ! : Part 5”

1. Pingback: [IdentityServer.v3, MembershipReboot, AngularJs, WebApi 2 and MVC : Mix It ! | {"@id":"cedric-dumont.com"}](https://cedricdumont.files.wordpress.com/2014/12/5-7.jpg)

[Blog at WordPress.com.](#) — [The Sequential Theme.](#)

© Follow

Follow “{"@id":"cedric-dumont.com"}”

Build a website with WordPress.com

