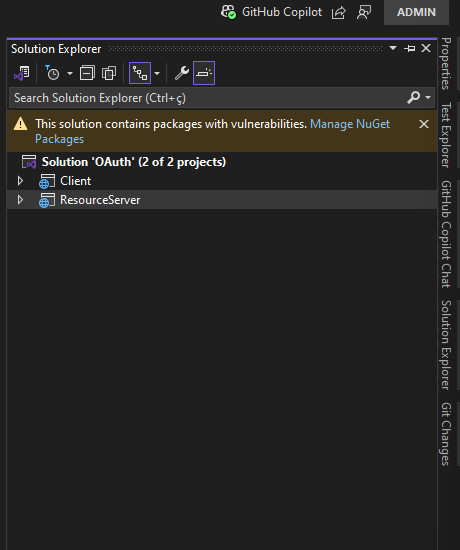
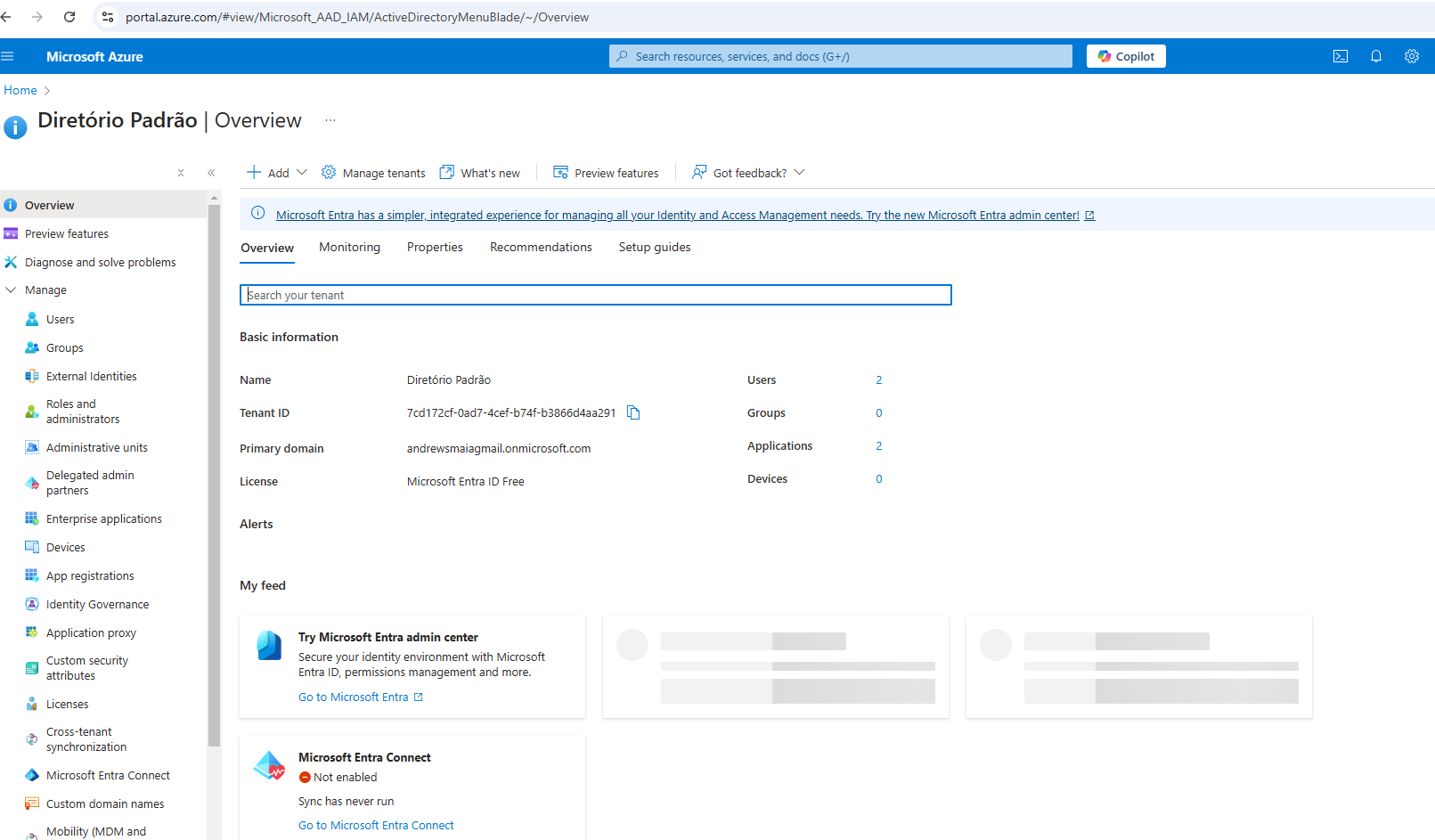
Temos aqui no Visual Studio dois projetos que representarão 2 personagens do fluxo authorization code: o Client e o ResourceServer

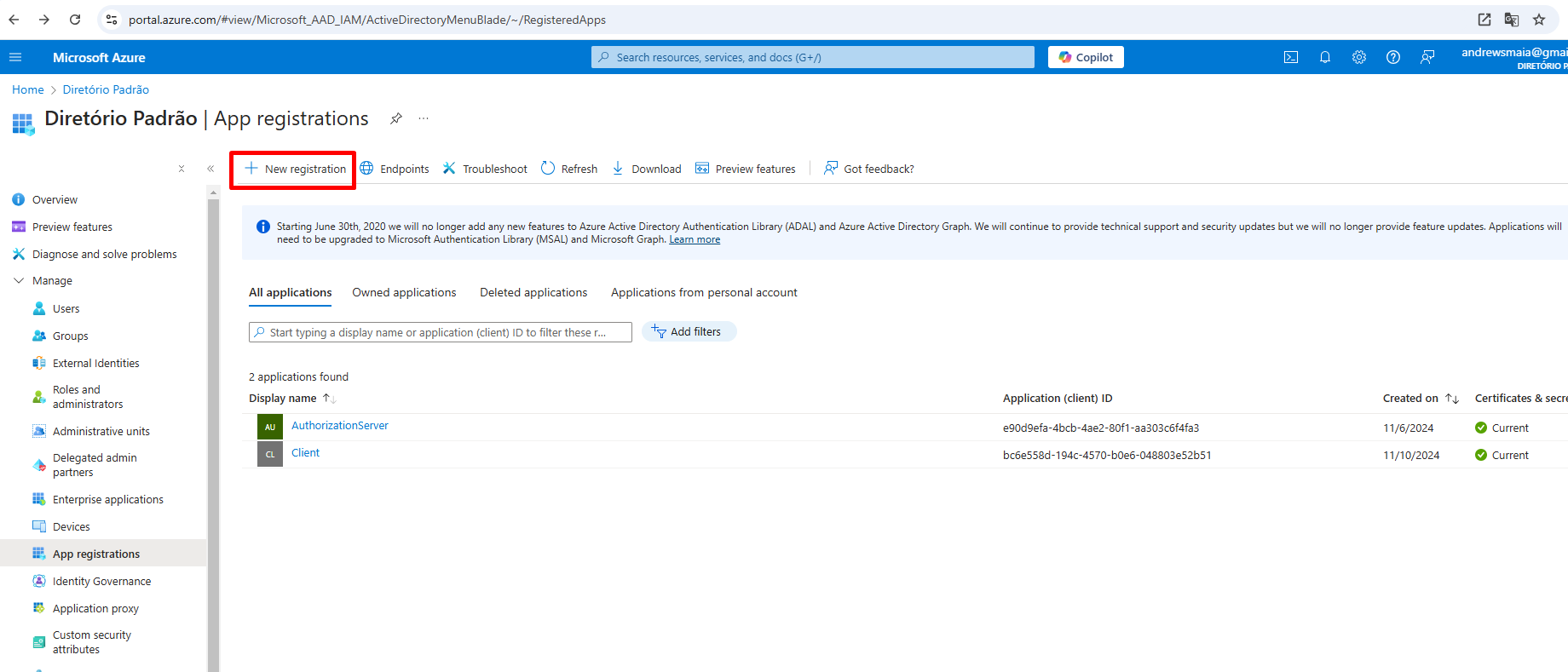
E o terceiro personagem vai ficar na nuvem. É o Azure Microsoft Entra ID e ele será o Authorization Server

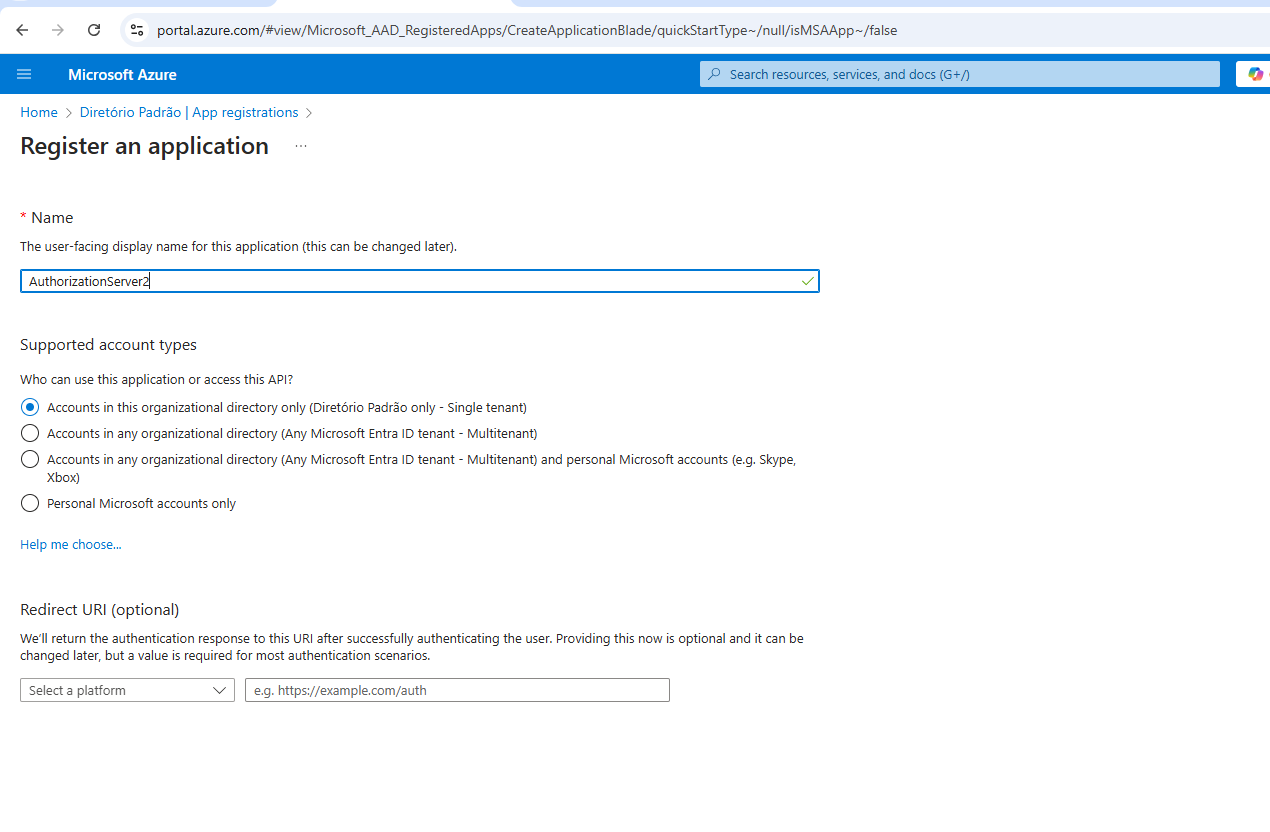


E é desse cara que vamos começar a falar primeiro.

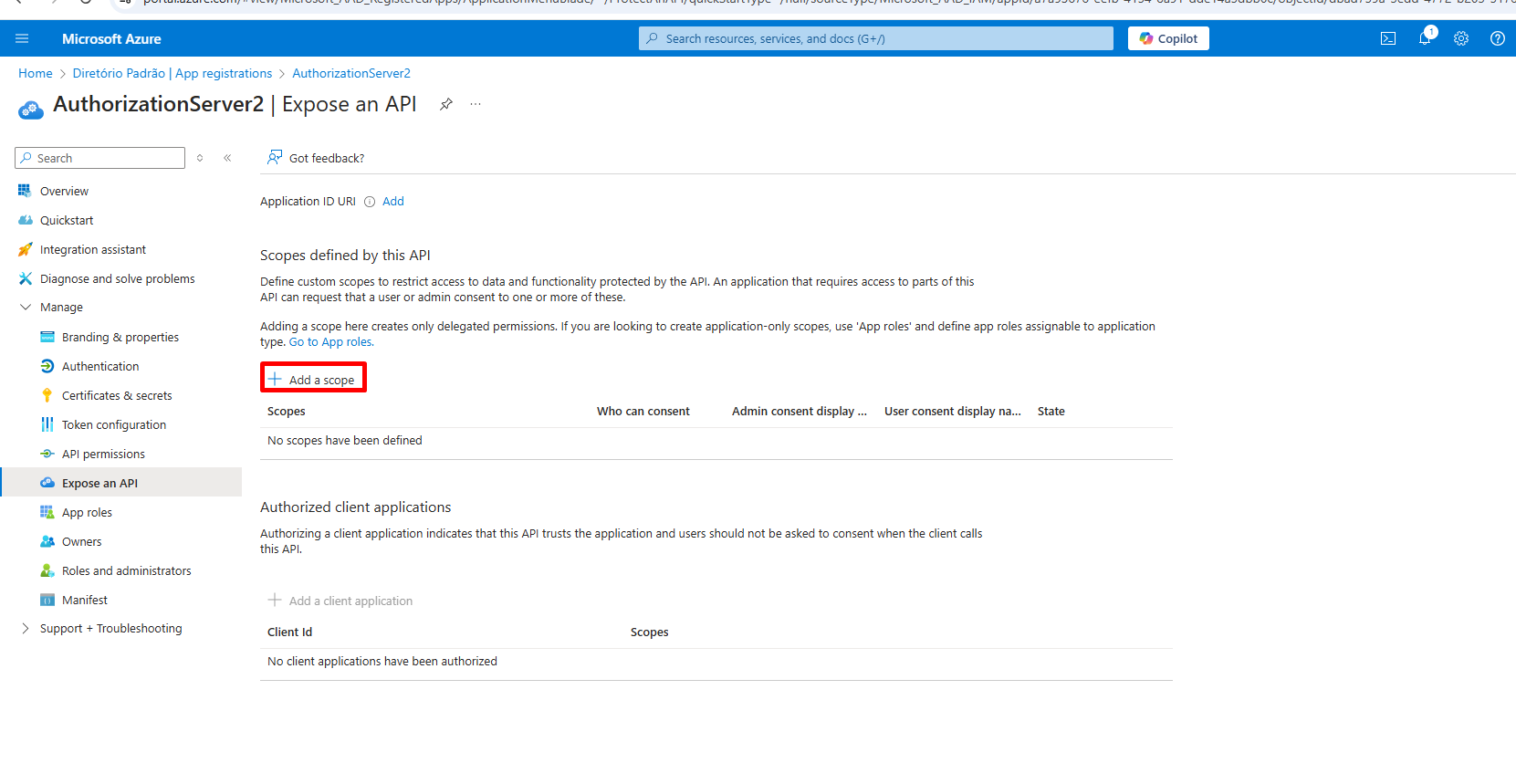
É o authorization server **que conhece os resourcers owners** (donos dos recursos) e **libera o acesso ou não aos recursos** conforme o consentimento do owner.

Então a primeiras coisa que vamos fazer no Microsoft Entra será registrar um app que irá conter o escopo que iremos utilizar.

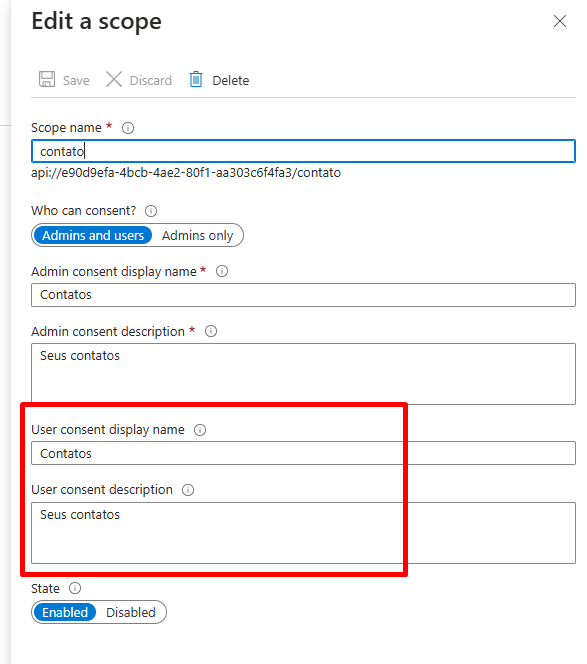




Depois criamos o escopo. No caso criaremos um escopo chamado contato, o qual caso o usuário consinta o acesso será possível o app client obter seus contatos.

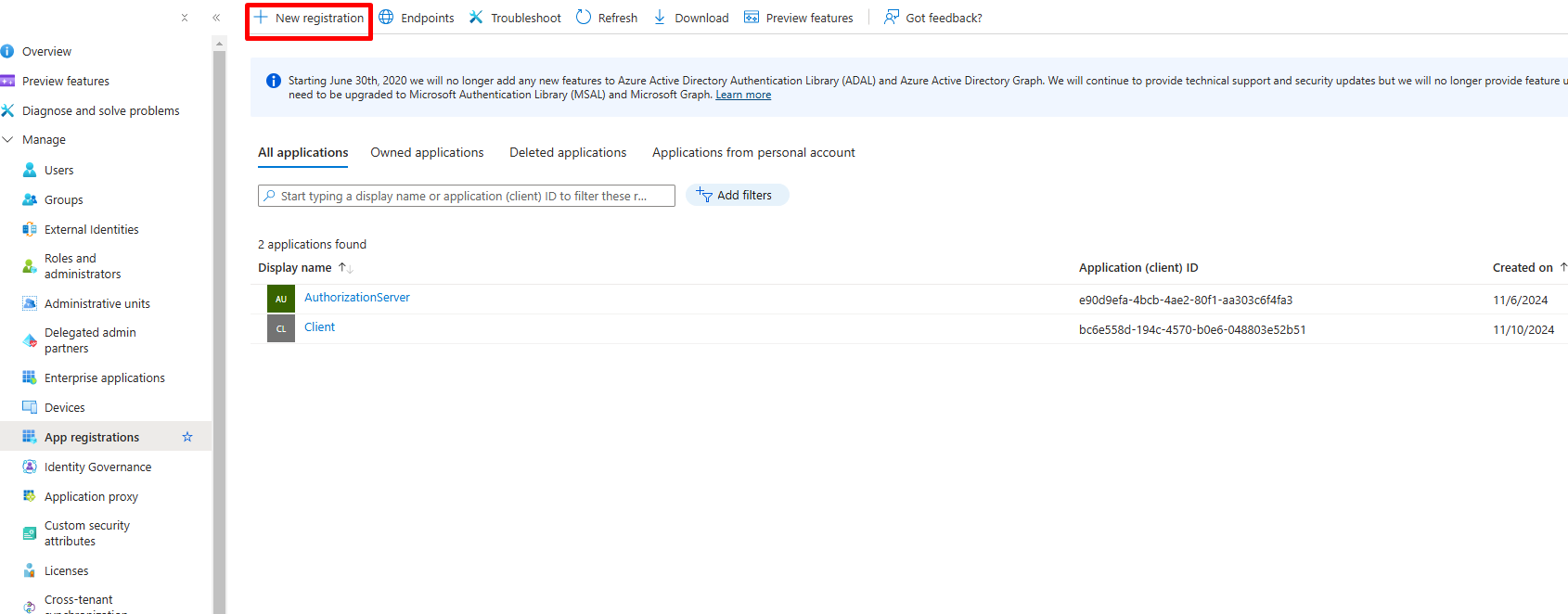


Repare que neste cadastro estão os campos “User consent display name” e “User consent description” que são os textos que irão aparecer na tela de consentimento

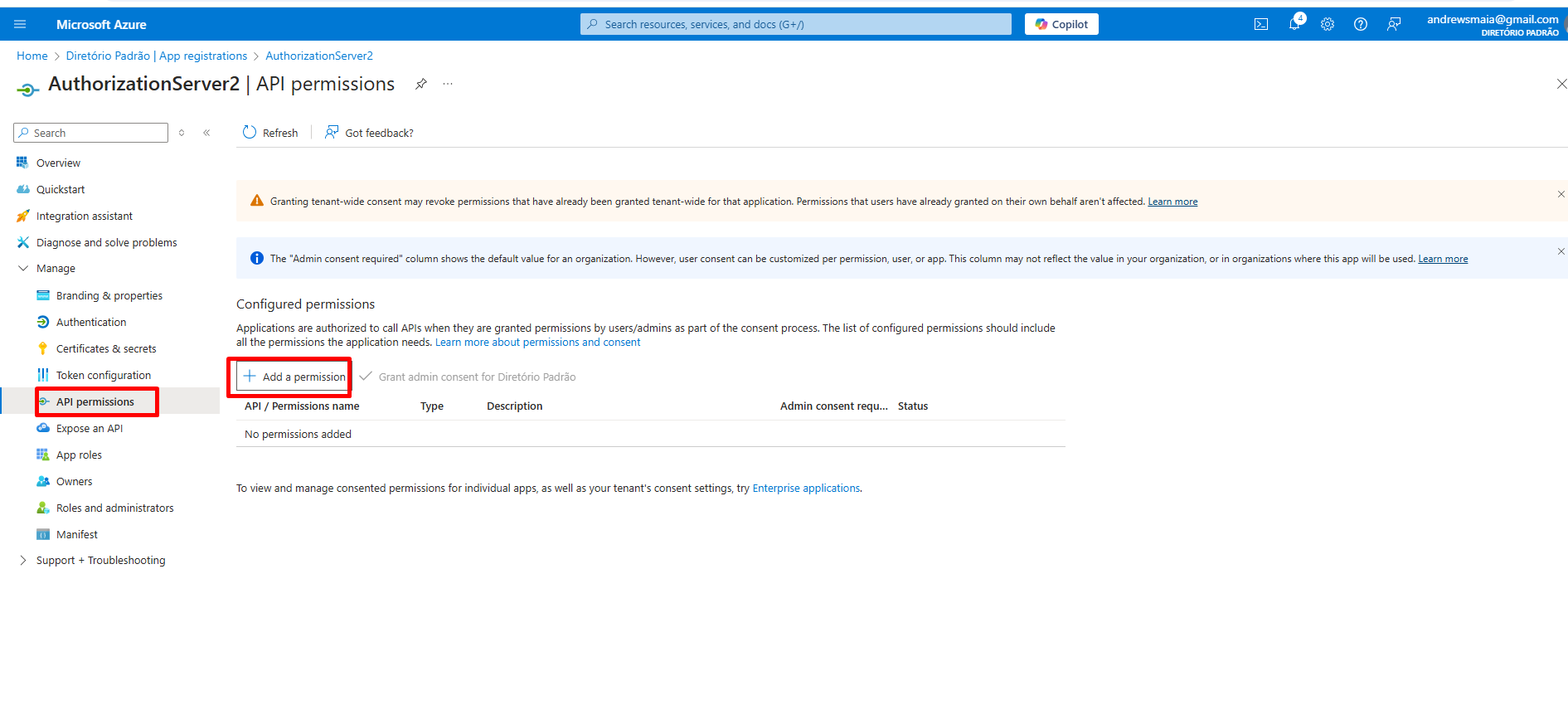


O segundo cara que precisamos configurar agora é o client. Lembra que precisa registrar o client no authorization server antes de começar a solicitar acesso a recursos?

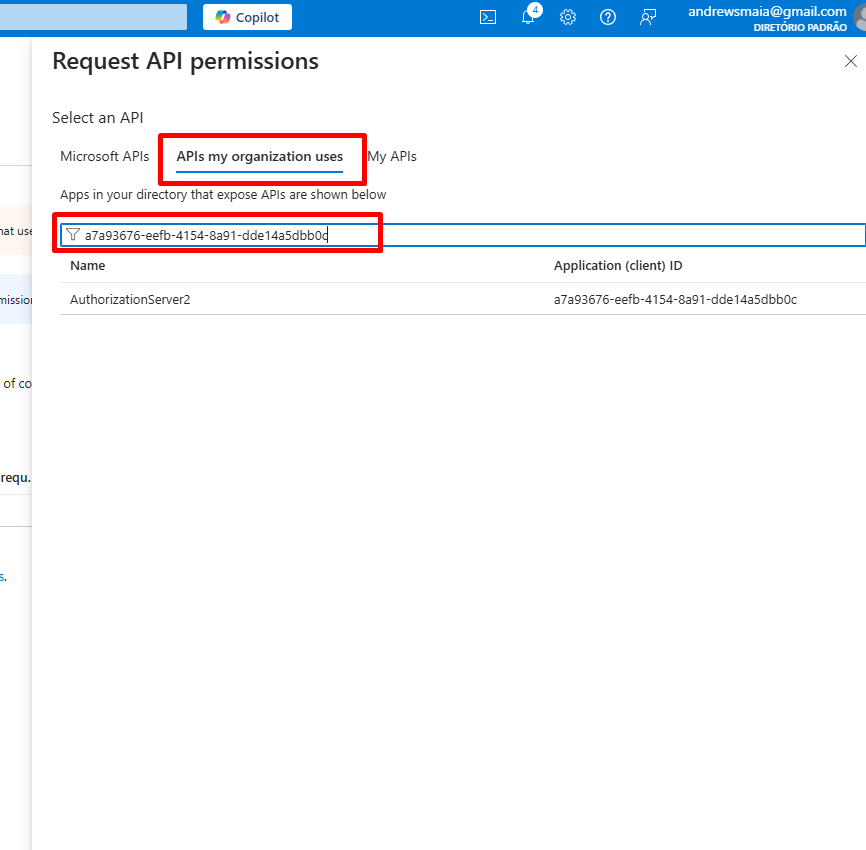
Primeiro registramos o app client:

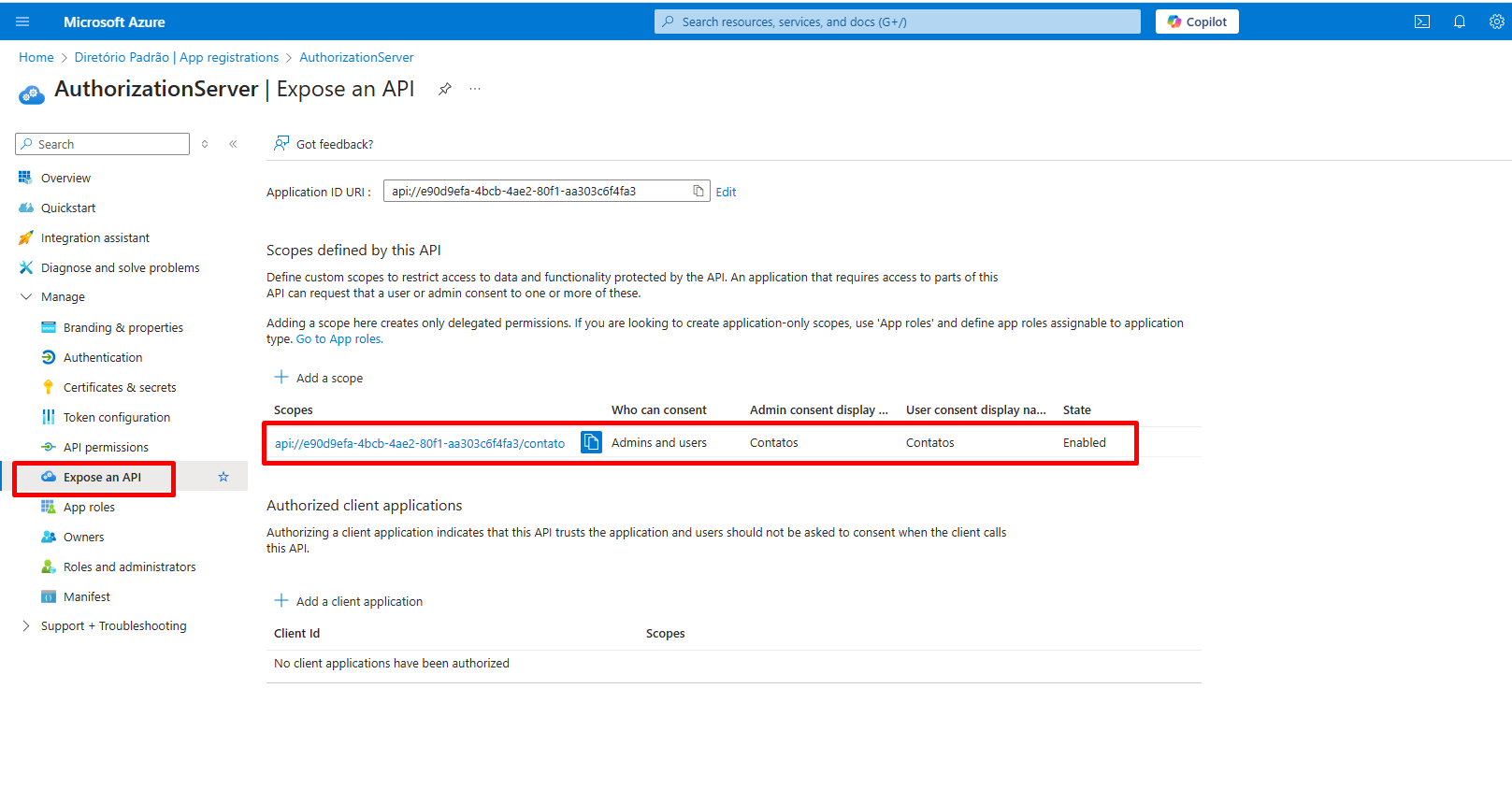
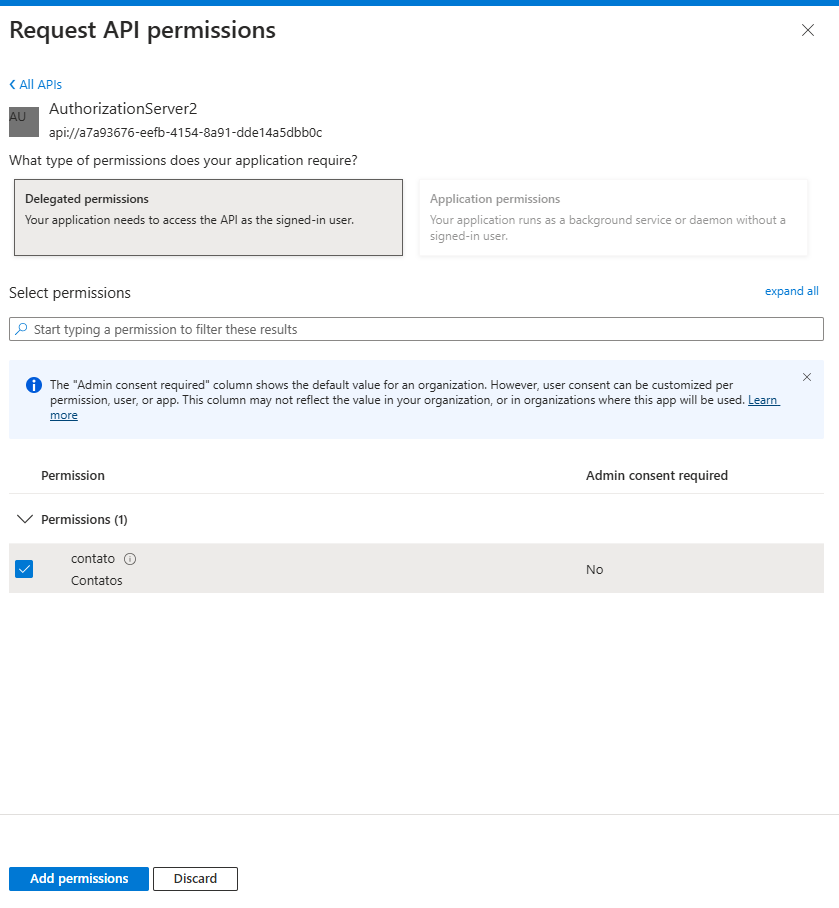


Dentro do App Client dizemos quais escopos ele pode solicitar acesso:

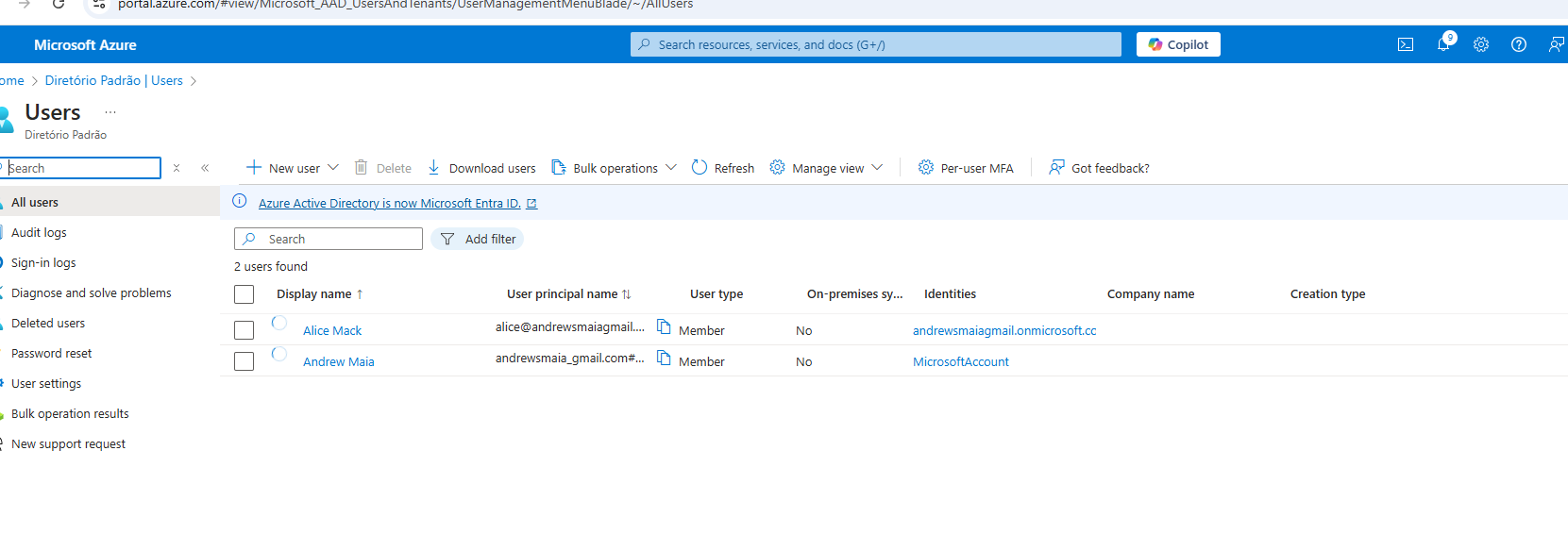


No caso será o escopo ‘Contato’ que definimos no app AuthorizationServer. Para encontrar esse escopo é necessário fornecer o id do app do AuthorizationServer:





Por fim precisamos criar os usuários que o authorization server vai lidar:

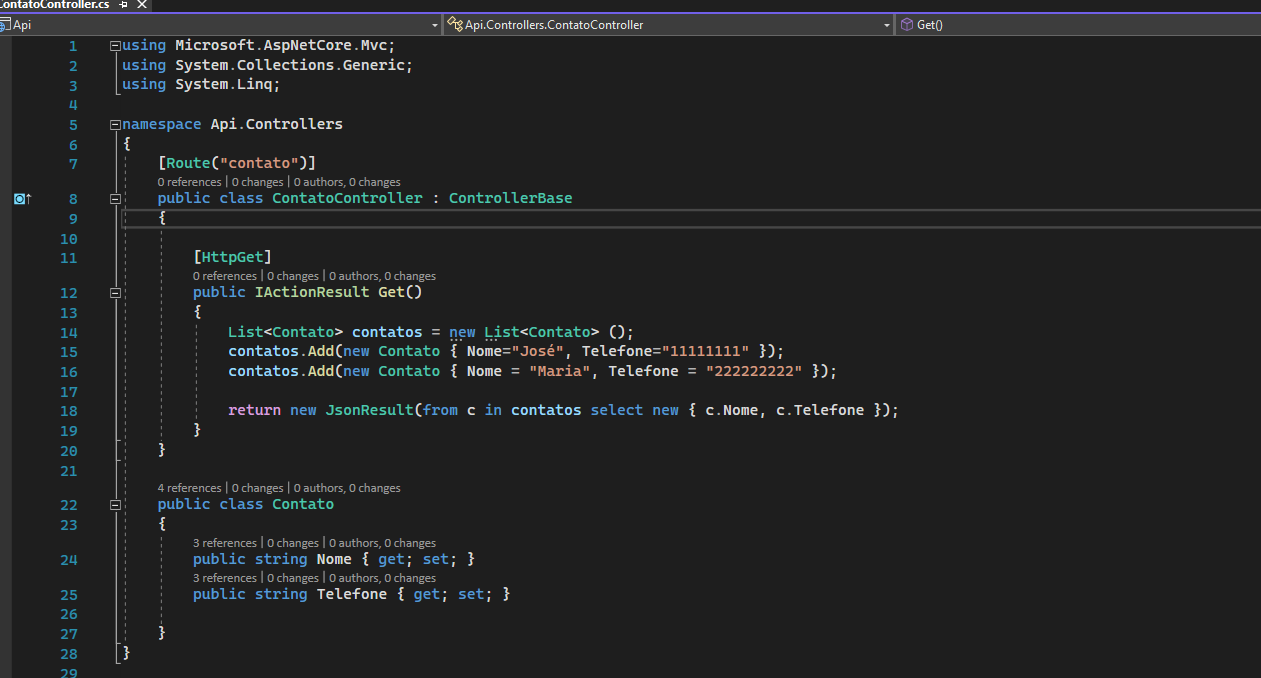


Aqui eu já criei um para teste chamado Alice Mack

De volta ao Visual Studio.

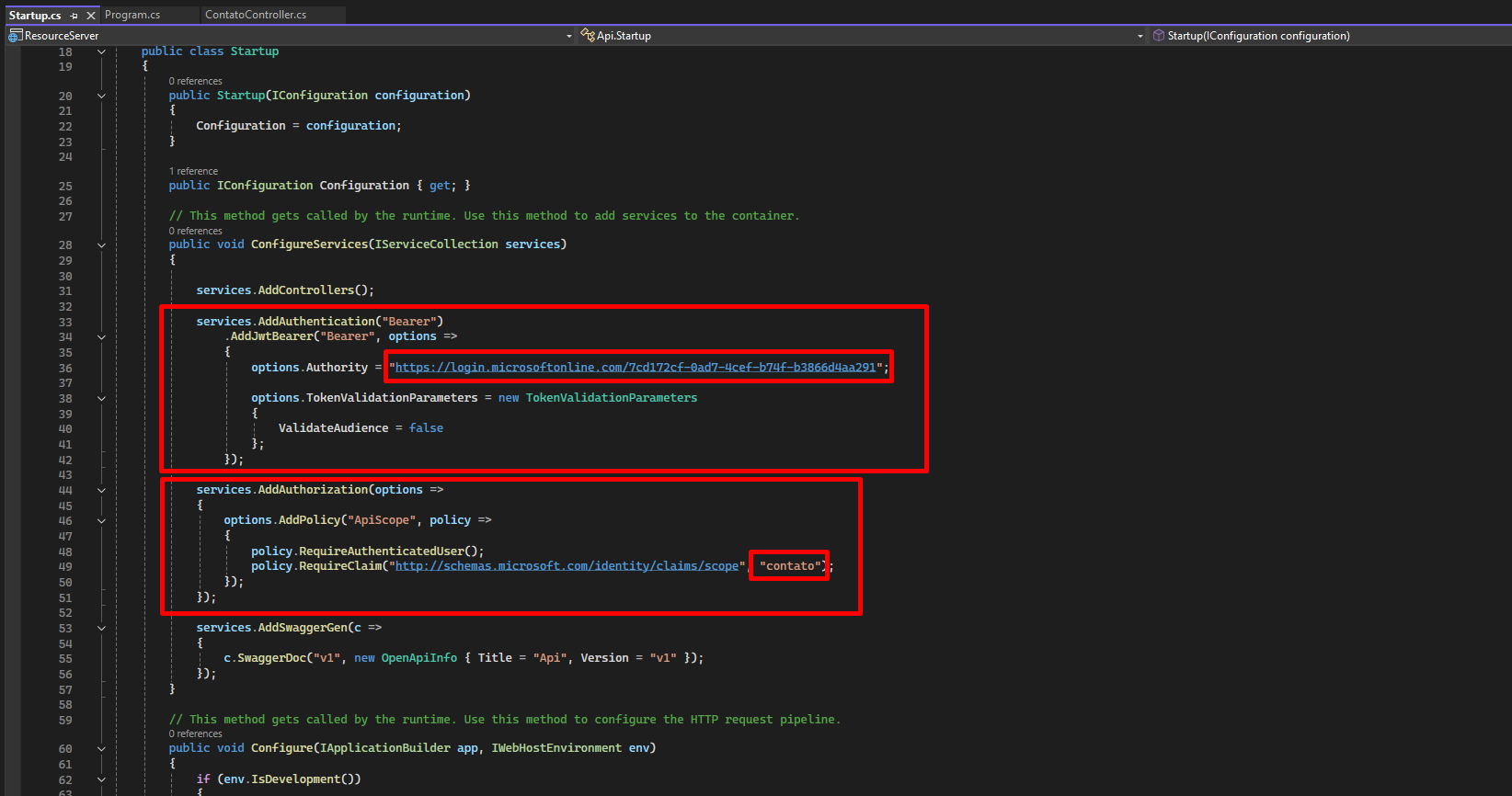
O próximo personagem é o Resource Server. Aqui estamos representando-o por um projeto de WebAPI.

Aqui temos uma API chamada contato que retorna contatos. São dados fakes e fixos que eu coloquei só para a apresentação. Ela retorna dois nomes e seus respectivos telefones



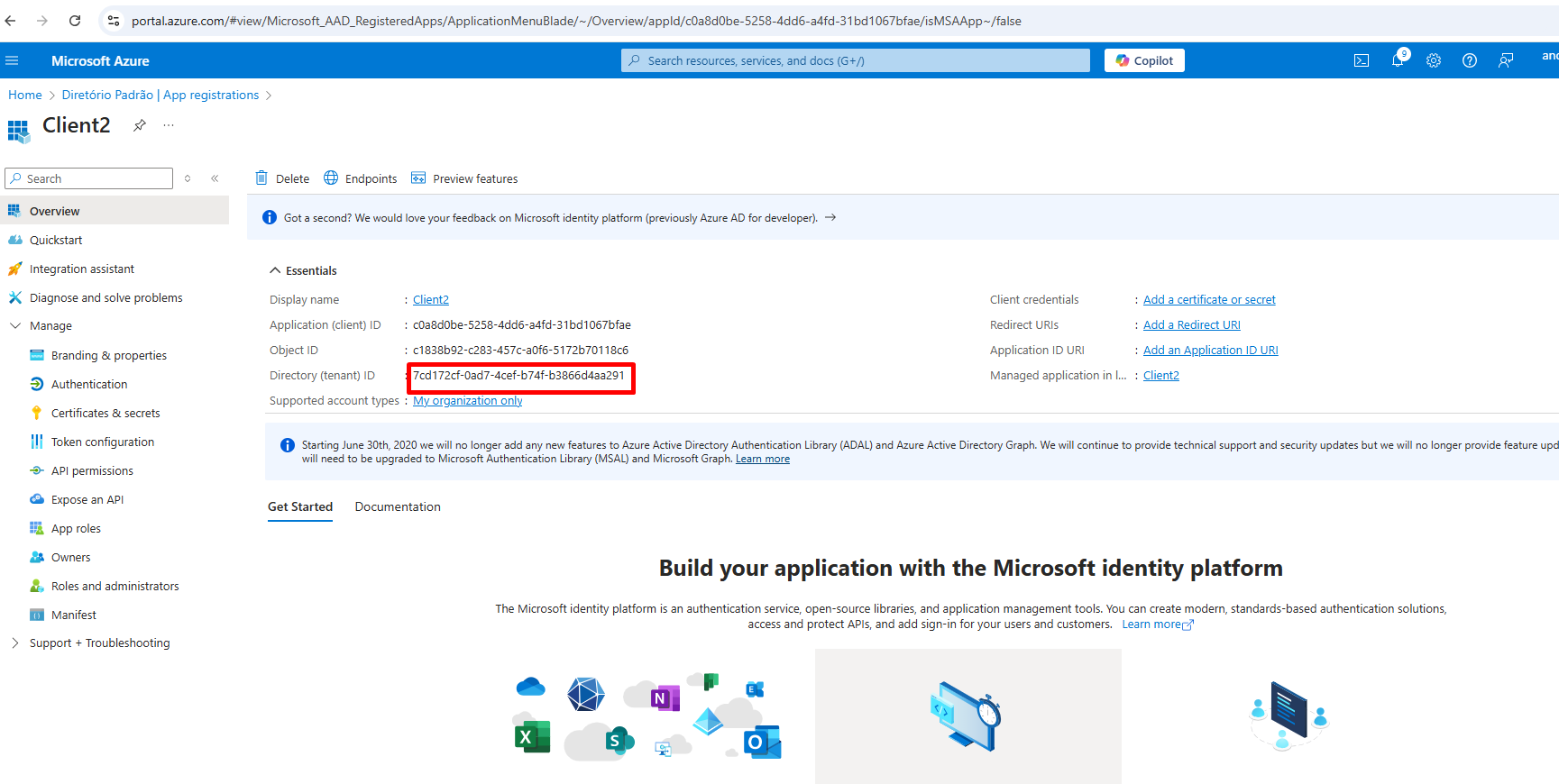
Essa API está protegida, ou seja, para acessar ela precisa de um token de acesso e o claim contato deve estar associado ao usuário em questão.

Essa proteção é feita por este trecho de código:



O principal aqui é a propriedade Authority que estamos apontando para o Microsoft Entra. Ou seja, é como se estivéssemos dizendo que quem é responsável por determinar quem pode acessar a API de contatos é o Authorization Server, que no caso é o Microsoft Entra.

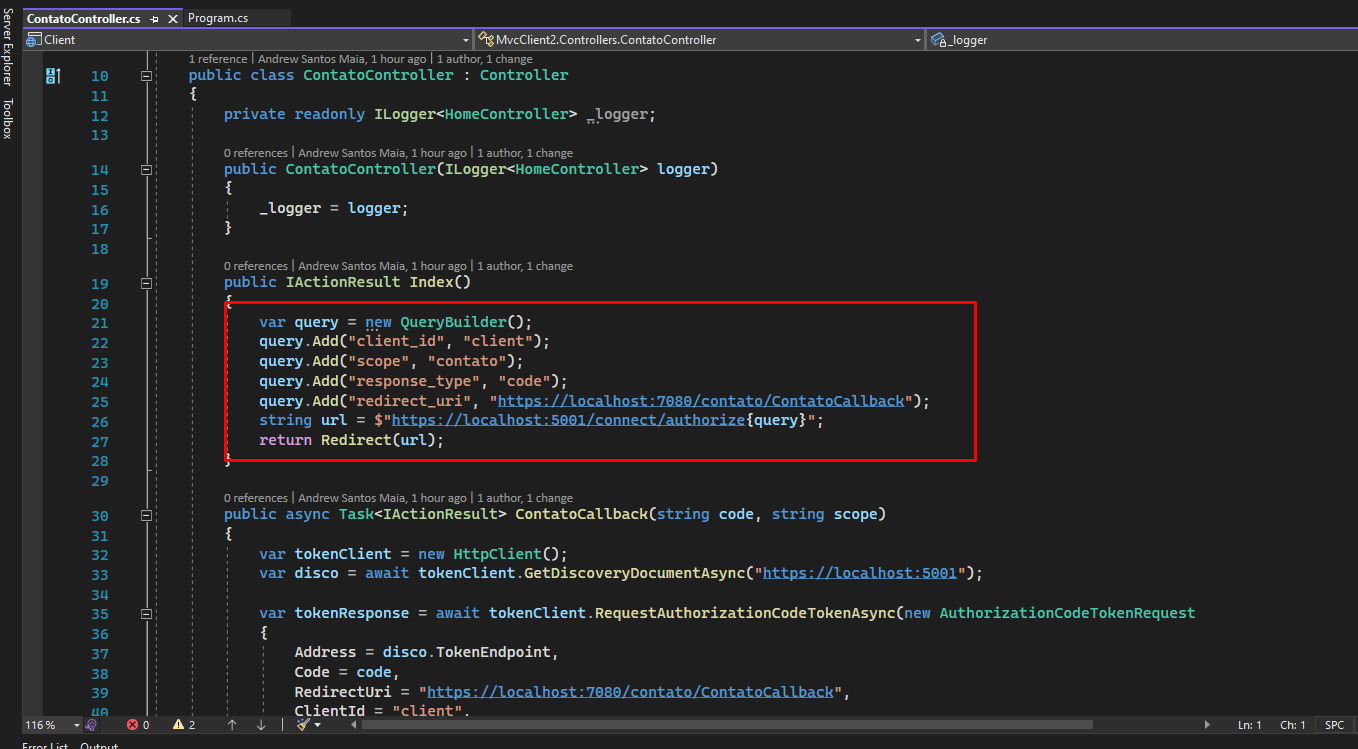
Esse endereço do Microsoft Entra é uma combinação entre a URL da Microsoft e o seu TenantId no entra que você pode obter aqui:



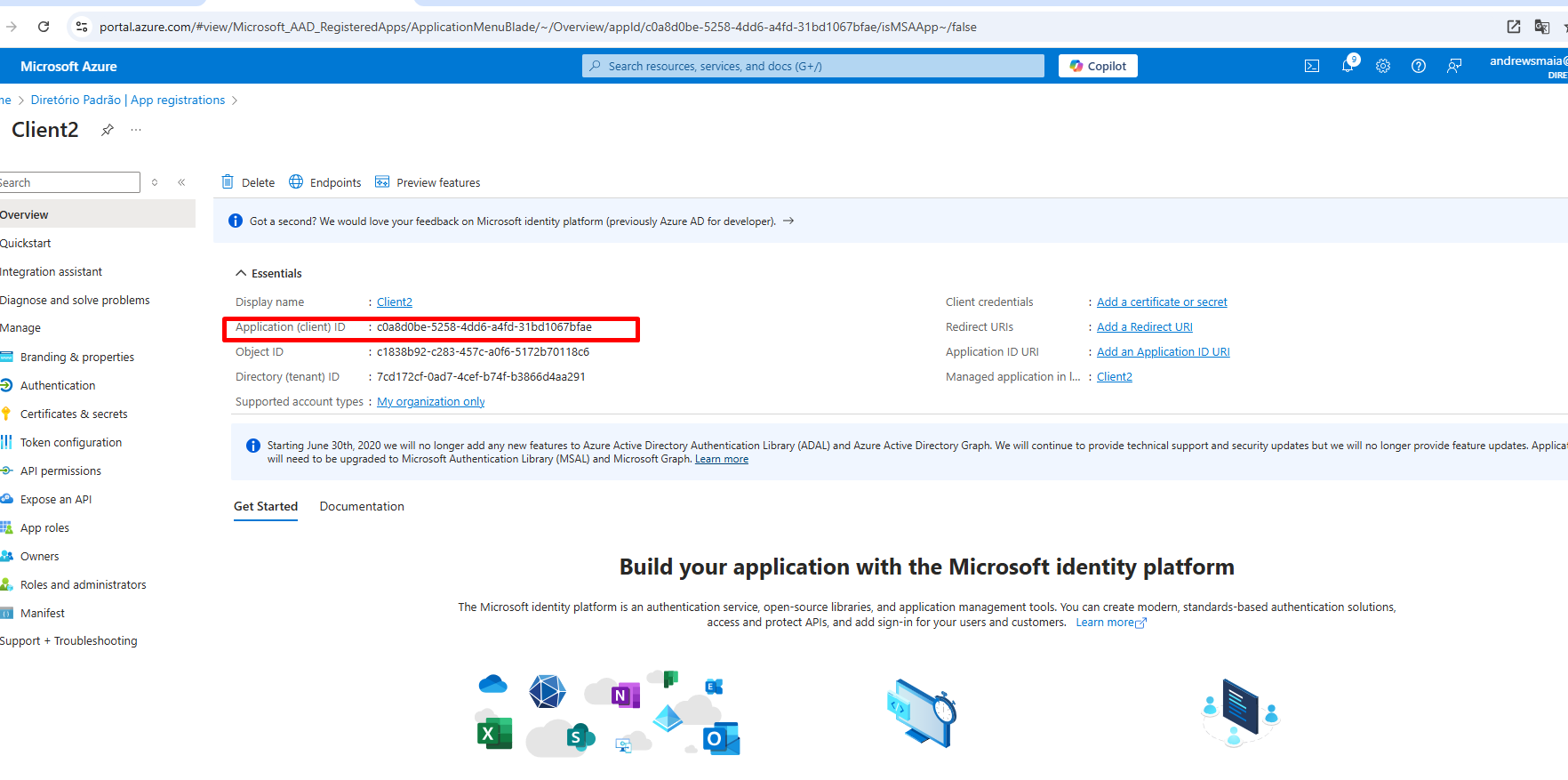
Por fim, o terceiro projeto é o Client. É um site que quer acessar os contatos de um usuário.

Aqui temos uma controller chamado Contato, que quando acesso-la, ela vai me redirecionar para o Microsoft Entra solicitando o meu consentimento para acessar os meus contatos.

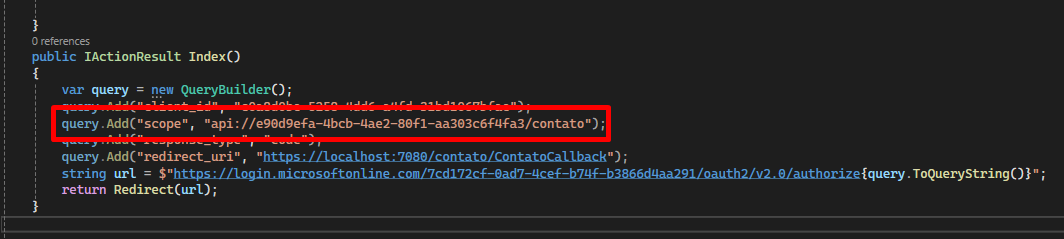
Este é um exemplo usando o fluxo authorization code, que vou explicar com mais detalhes agora.



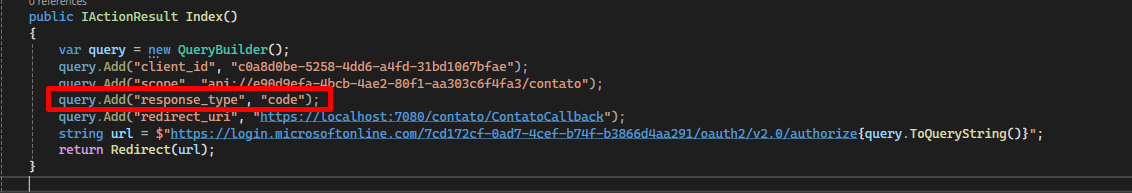
Aqui precisamos primeiramente o clientId. Para obter o clientId vá até a página de overview do app criado para client no Microsoft Entra:



Depois precisamos fornecer os escopos que queremos acesso. No caso contato:



O response\_type que é o fluxo de autorização que será do tipo code:

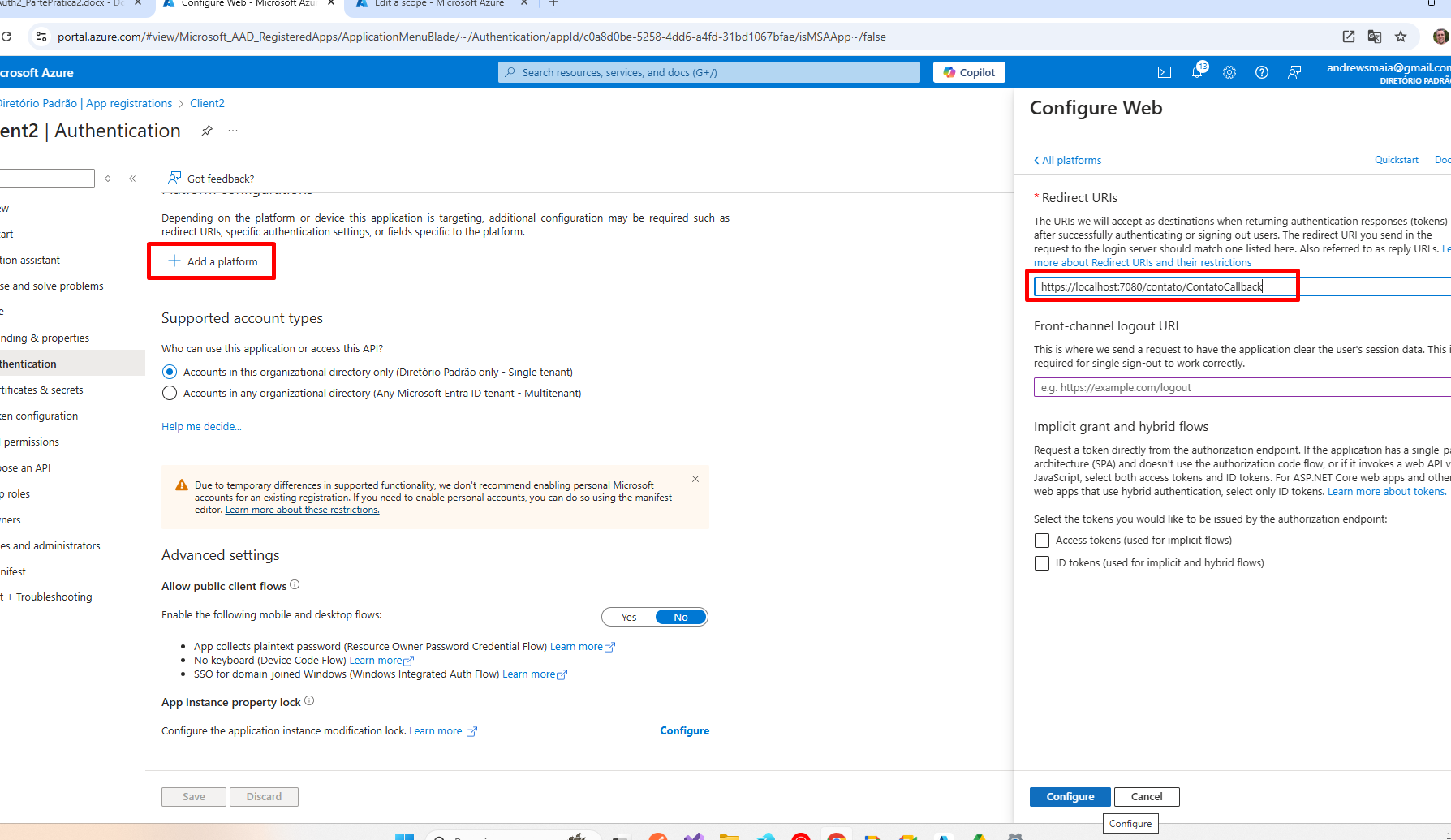


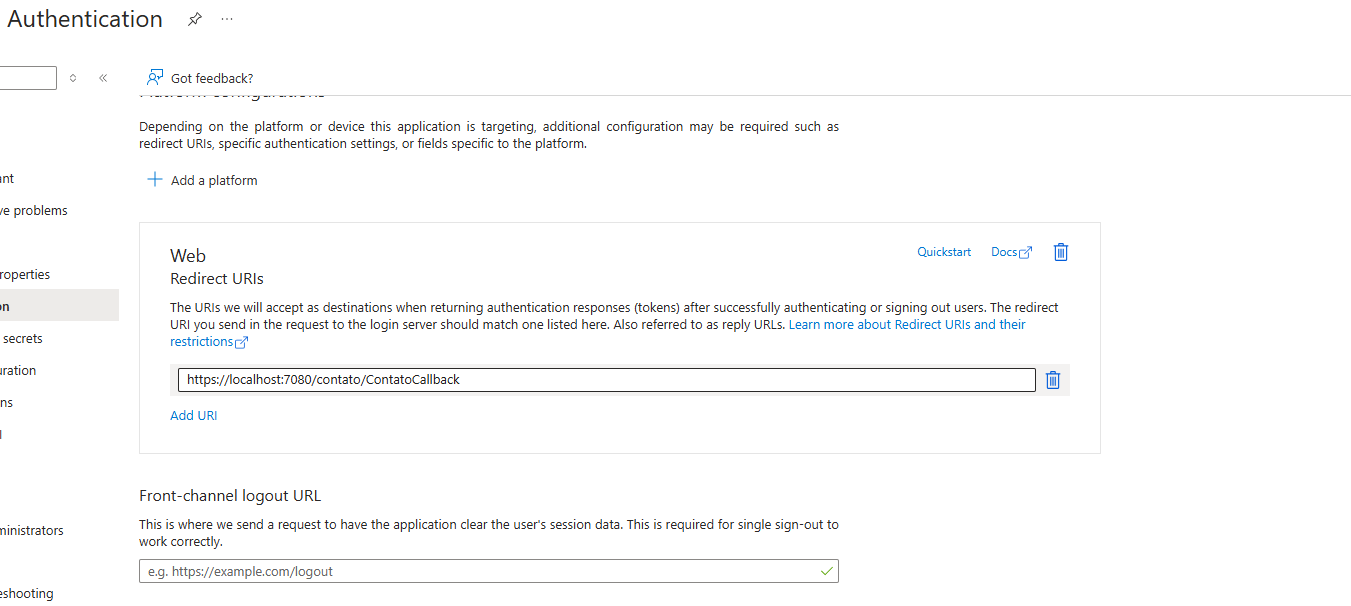
E a URL de redirecionamento. A url de redirecionamento é utilizada pelo o authorization server para retornar para o client depois que o usuário autoriza o acesso ao recurso.

No caso, estou usando um outro endpoint dentro da própria controller de contato chamado ContatoCallback.

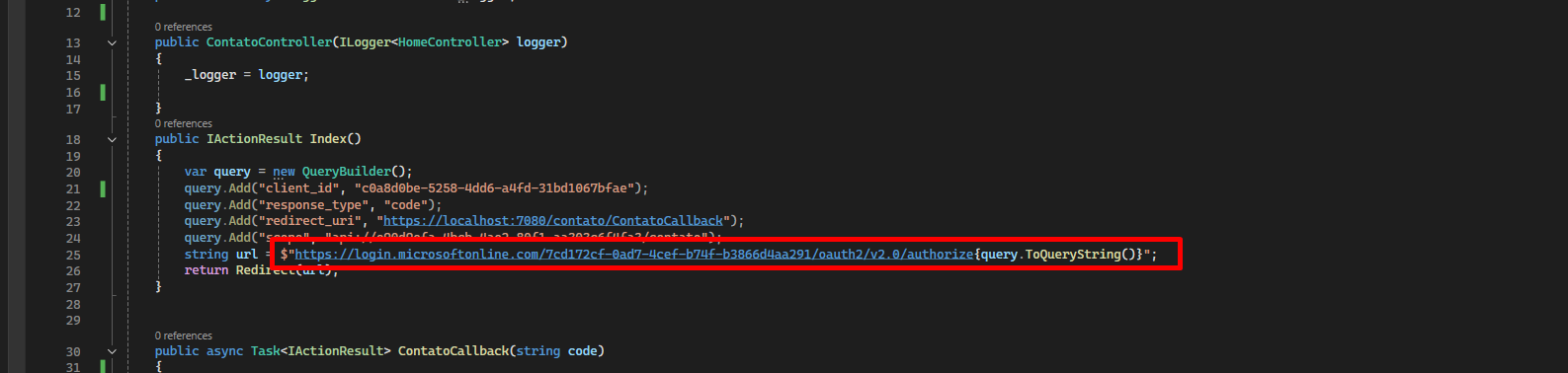


Importante a gente precisa cadastrar essa URL de redirecionamento no cadastro do app client no Microsoft Entra:



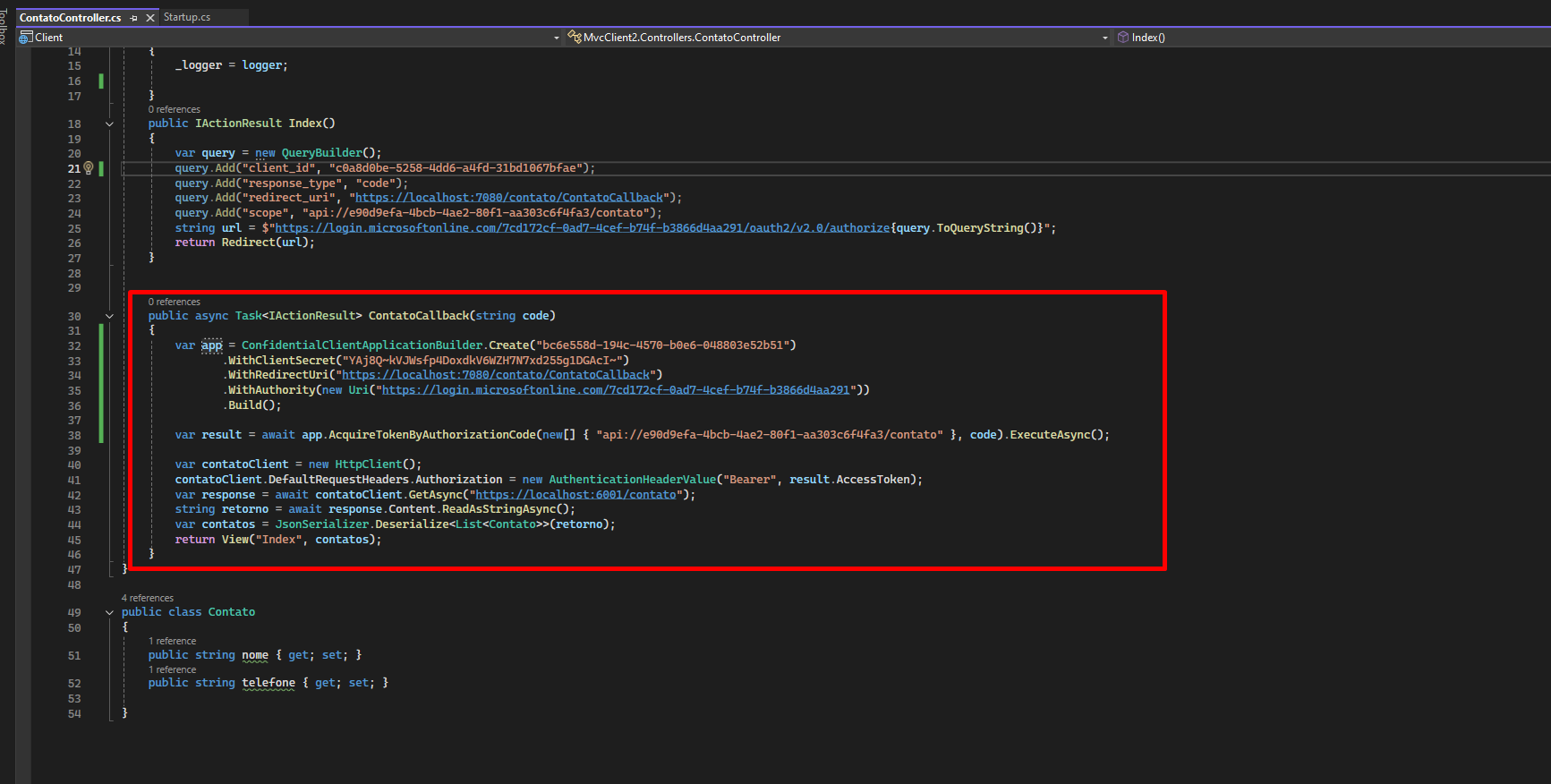


Por fim encaminhamos esses parâmetros via querystring para o endpoint de autorização do Microsoft Entra:



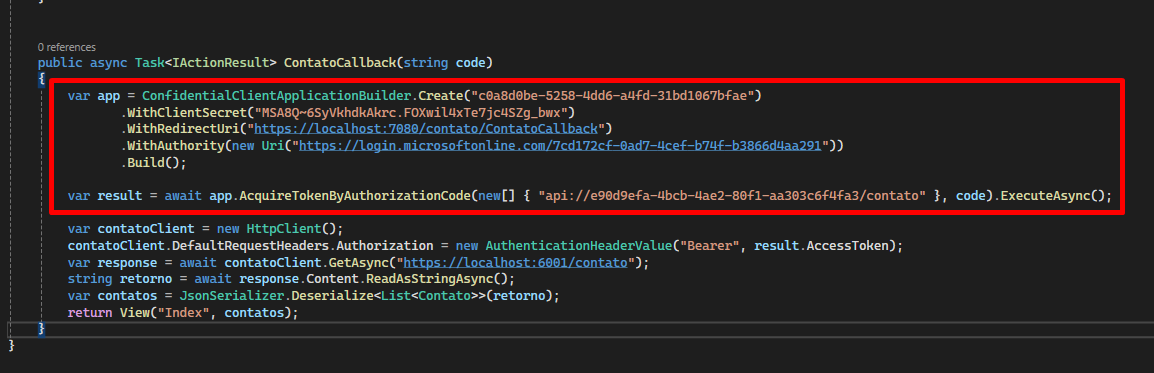
Repare que novamente precisamos usar o tenantId do Microsoft Entra.

O endpoint ContatoCallback é para onde authorization server redireciona o usuário de volta para o client. Repare que o authorization server retorna um code na query string. Esse code faz parte do fluxo authorization code, é através dele que iremos obter o access token para assim conseguir acessar a API de contatos:



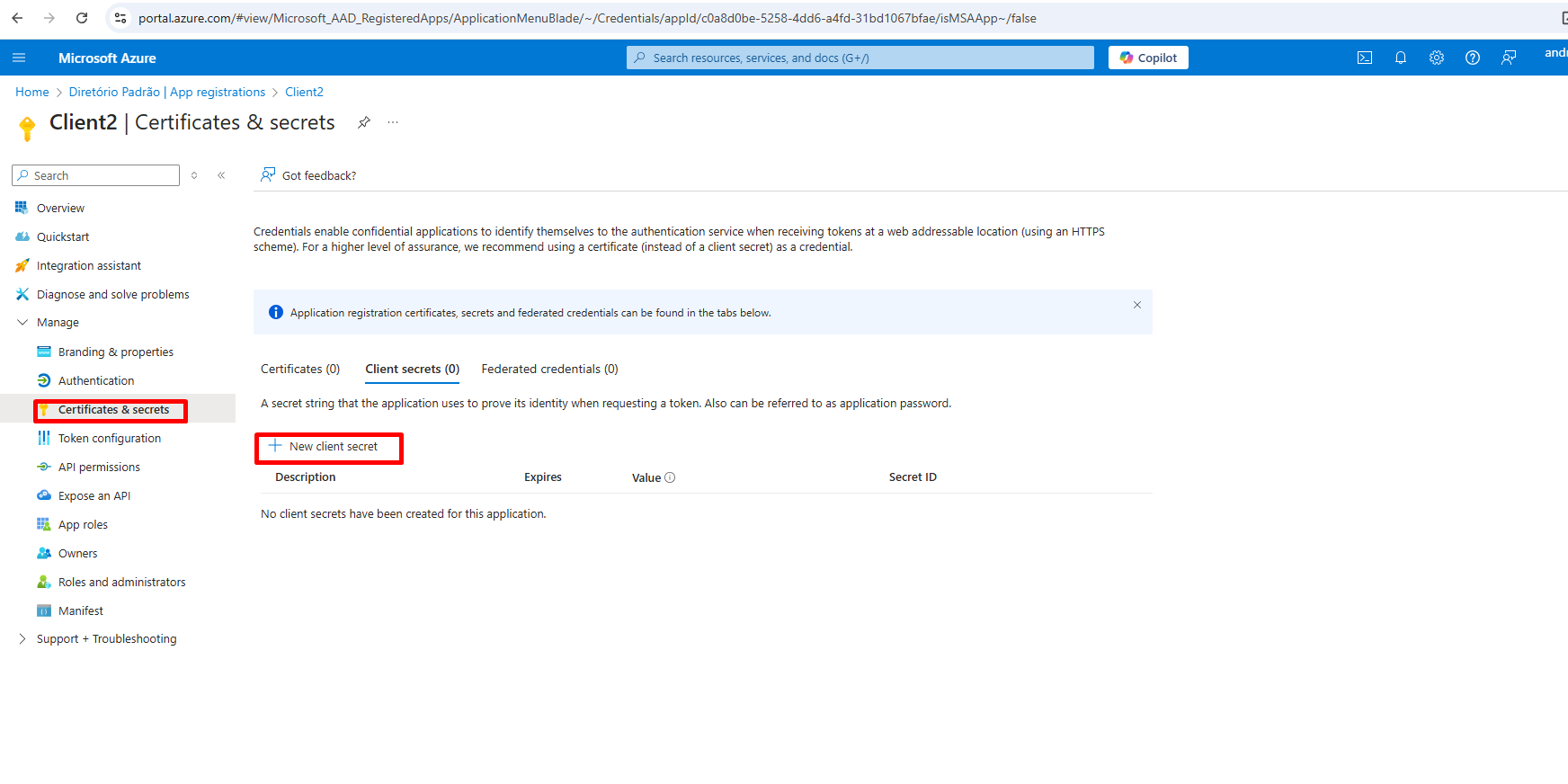
Depois que obtemos o code da query string, vamos de novo no authorization server para obter o **authorization token** só que sem a participação do usuário diretamente, de server para server, o que nós chamamos no mundo OAuth2 de **backchannel**.

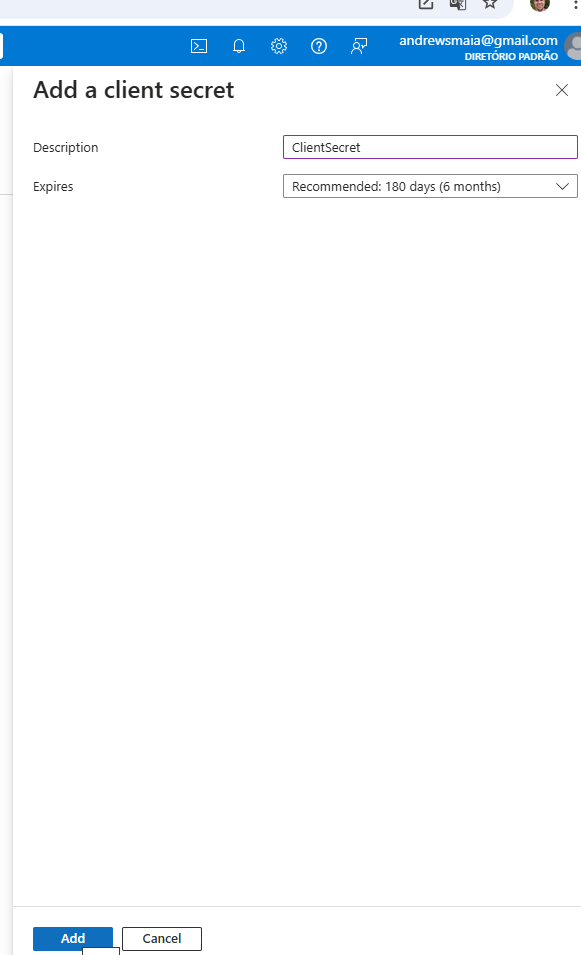
Aqui usei uma lib chamada Microsoft.Identity.Client:

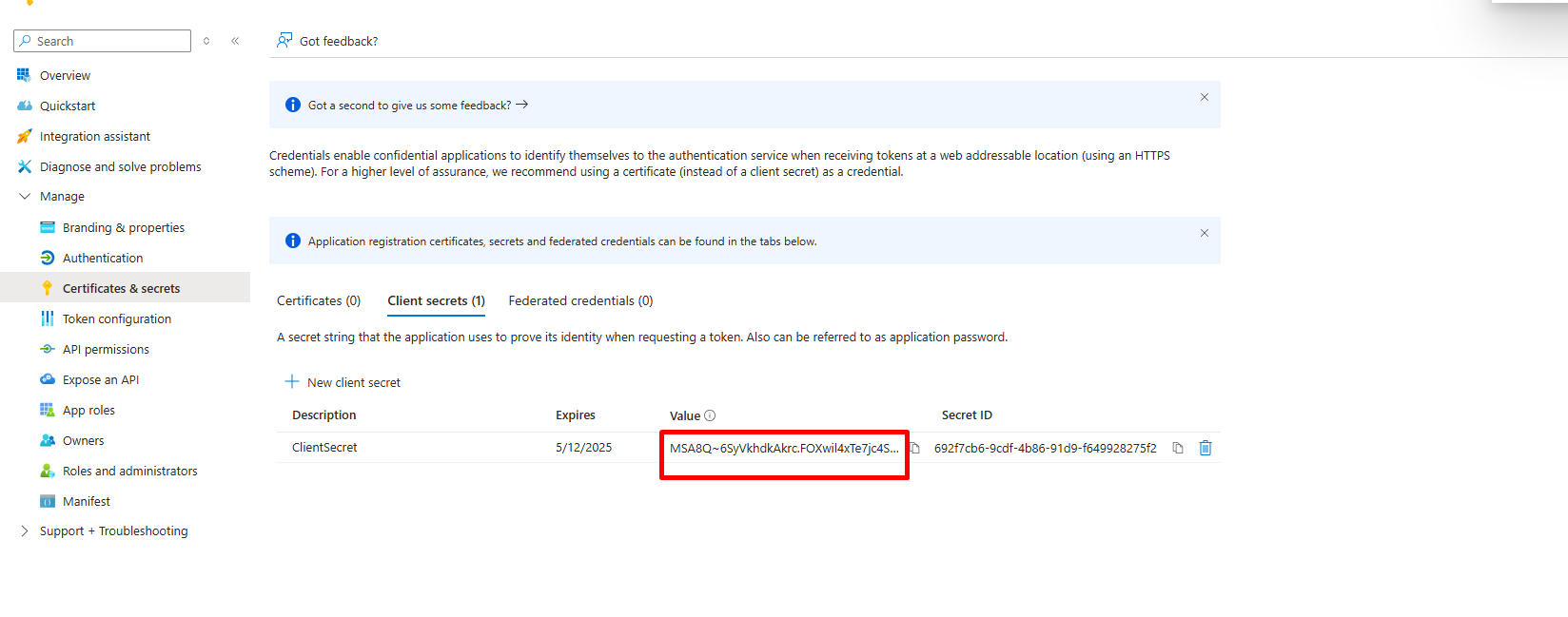


Nessa lib primeiro precisamos fornecer o clientId, o qual já temos.

Segundo precisamos fornecer o clientSecret. Para gerar o client secret devemos voltar ao Microsoft Entra e no cadastro do app client e gerar esse secret:

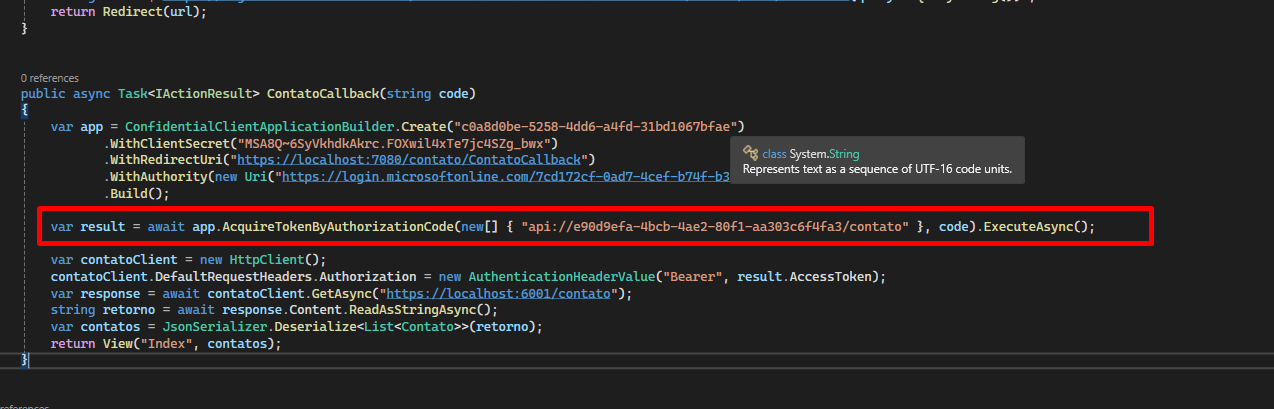




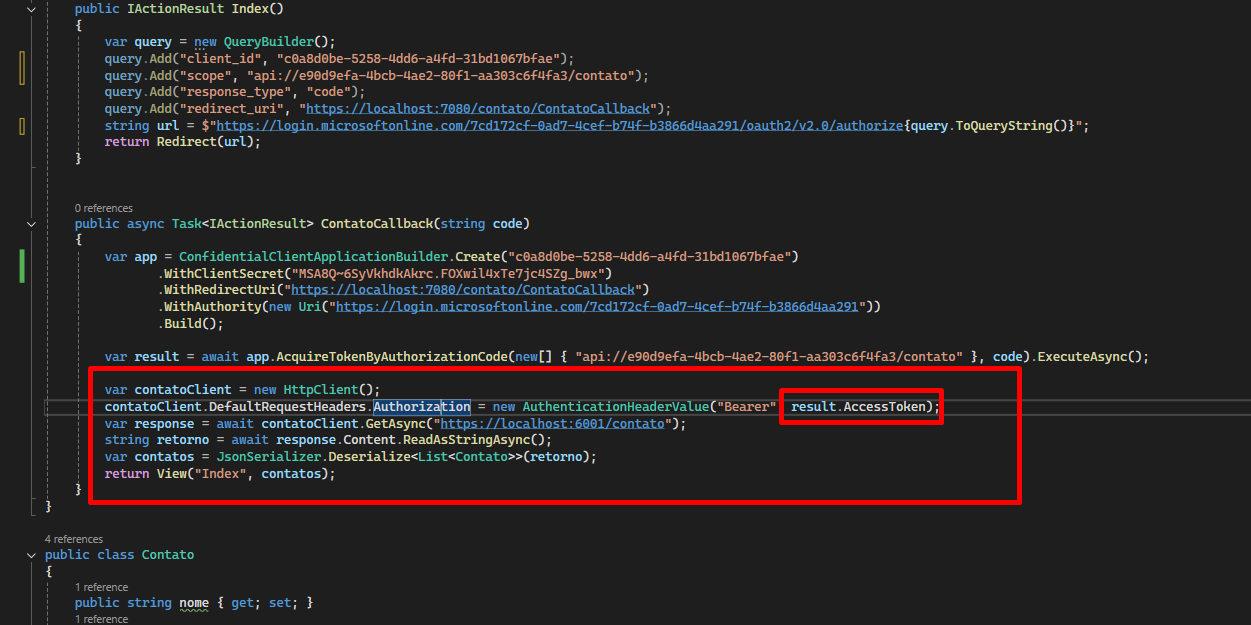


Depois a redirect URI, a qual já temos também.

E por fim o endereço da autoridade, no caso a url do microsoft entra com o meu tenantId

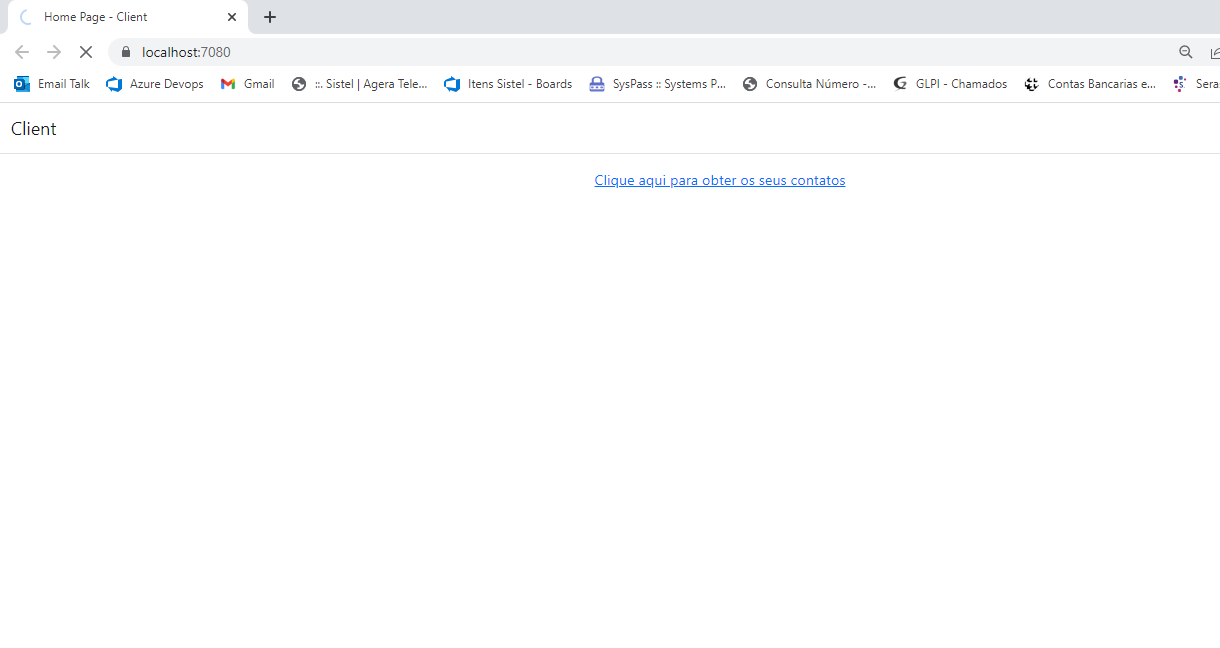
Após a configuração da lib, fazemos uma chamada solicitando o access token para o escopo contato:

Com esse token podemos utilizar na chamada para o terceiro personagem, o resource server, para obter os contatos do usuário:

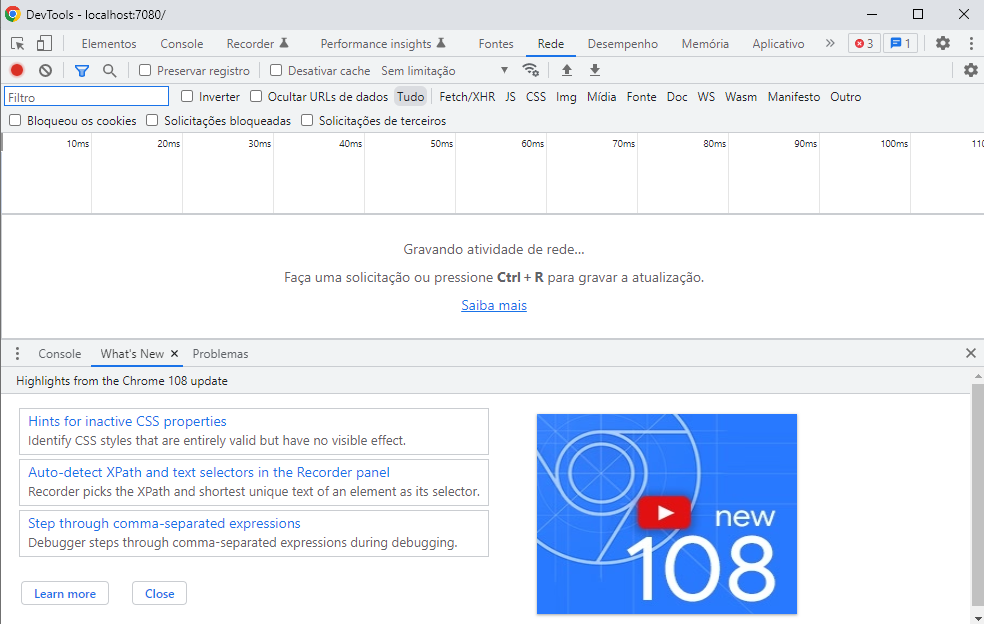


Vamos ver na prática.

Acessamos o site do Client que está rodando na porta 7080:



Eu vou deixar o DevTools do chrome aberto para vermos o que ocorre durante a comunicação entre o authorization server e o client:



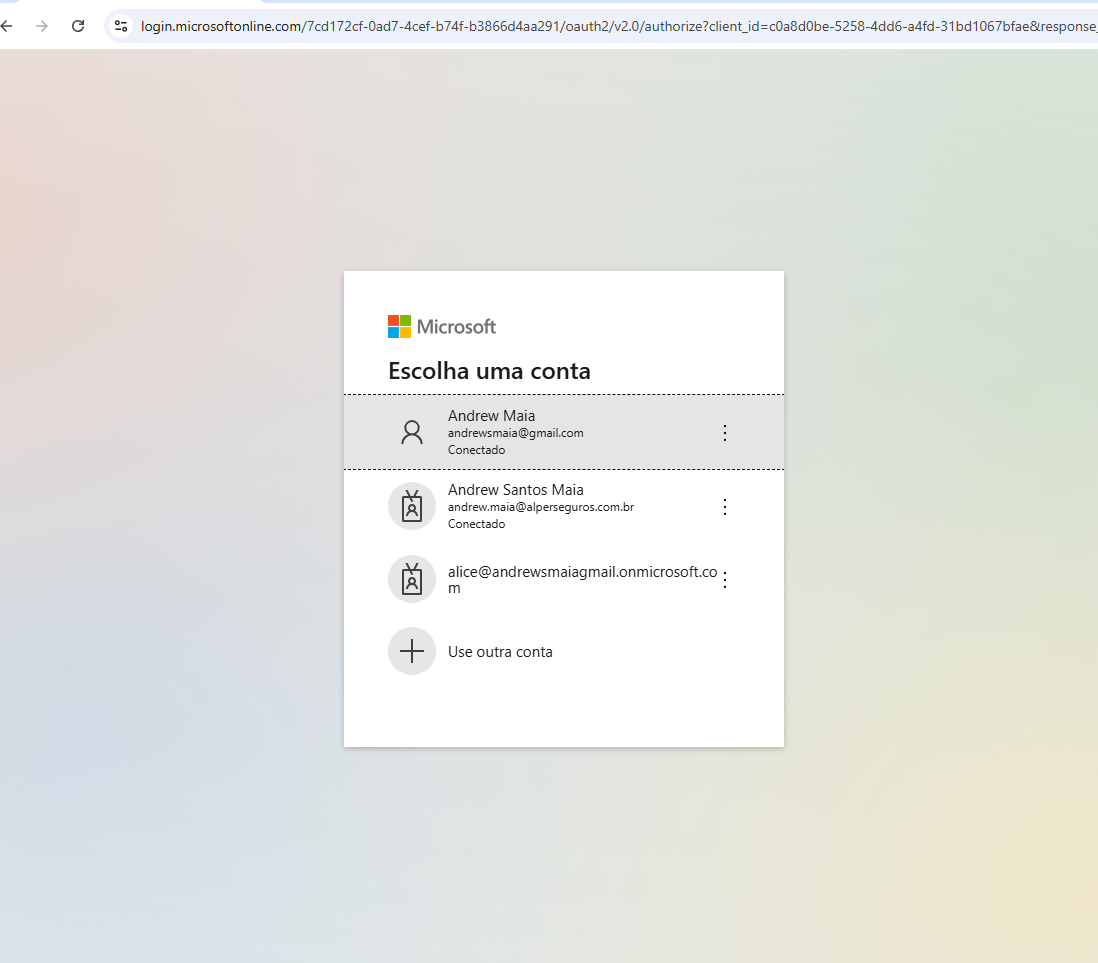
[Ir debugando o código]

Ao clicar no link para obter os contatos, somos redirecionados para o Microsft Entra.

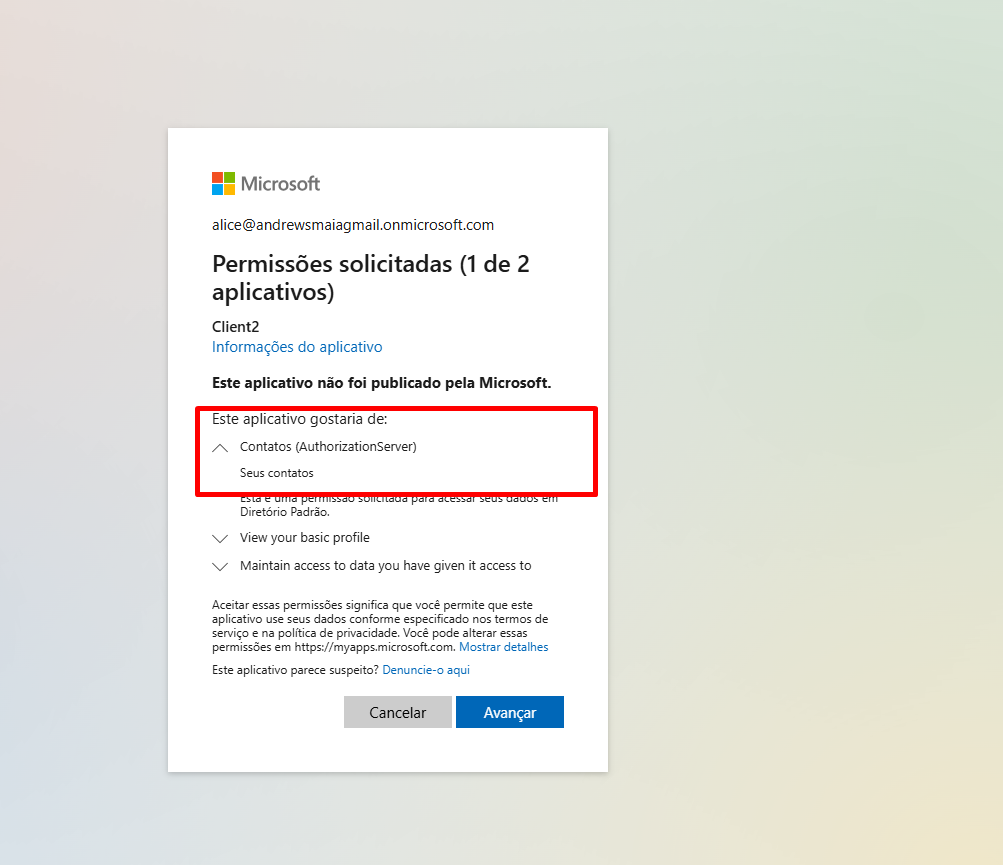
Primeiro o usuário é solicitado a fazer o login:

usuario: alice@andrewsmaiagmail.onmicrosoft.com

senha: ihKa^~!8S\*17



Depois vem a tela de consentimento onde usuário irá autorizar ou não o acesso ao escopo solicitado pelo client. Repare que o que aparece aqui para o resource owner é o nome **amigável** do escopo que definimos no Microsoft Entra:

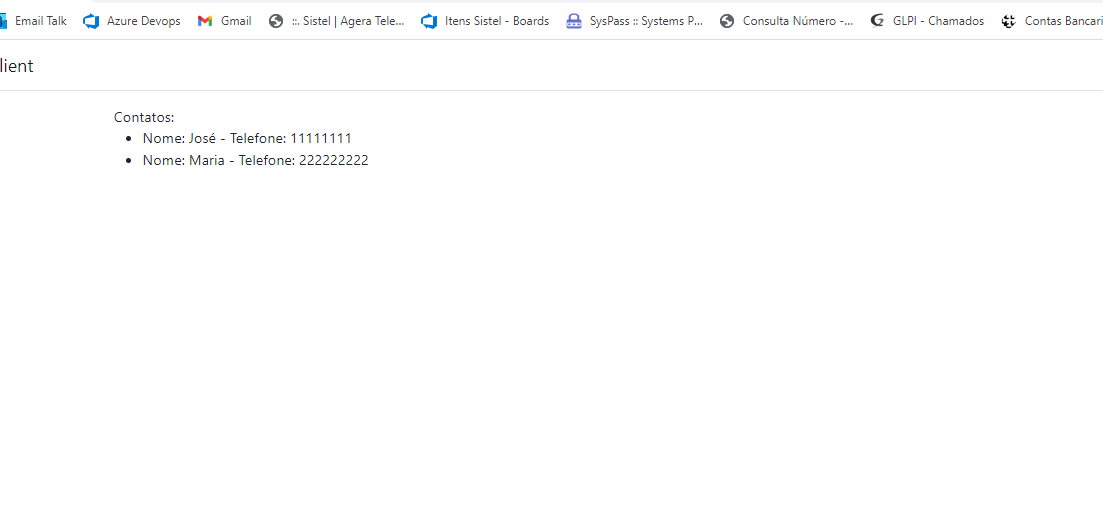


[Ir debugando o código]

Com o consentimento liberado o authorization server (Microsoft entra) retorna o code para o client.

A partir do code ele consegue fazer uma nova requisição para o authorization server para obter o token de acesso via **backchannel** .

Depois ele faz uma requisição para a API de contatos do resource server que está rodando na porta 6001 com o token para obter os contatos do resource owner:



No Devtools podemos observar o que ocorreu:

Primeiro o usuário é redirecionado para o endpoint authorize do Microsoft Entra. Depois do consentimento, o Microsoft Entra redireciona o usuário para o endpoint redirect uri do client onde o client recebe o code. Depois não podemos observar mais nada aqui no chrome porque todas as requisições ocorrem no backchannel , ou seja, de server para server.

