**Why an OpenID Connect server?**

Adding an OpenID Connect server to your application **allows you to support token authentication**. It also allows you to manage all your users using local password or an external identity provider (e.g. Facebook or Google) for all your applications in one central place, with the power to control who can access your API and the information that is exposed to each client.

[OpenIddict](https://github.com/openiddict/openiddict-core) is an open source framework for ASP.NET Core which allows you to easily implement an [OpenID Connect](https://openid.net/connect/)server.

JWT Middleware + OpenIddict/IDS4.

You will need to have some sort of OpenID Connect authorization server such as OpenIddict or IDS4 (or a commercial offering like Auth0, Okta, etc) which will be used to authorize the users and issue an id\_token + access\_token. These tokens can be stored in local storage in the browser.

Your Angular app will then need to pass the access\_token in the Authorization header with requests being made to your ASP.NET Core API. Inside the ASP.NET Core API you can use JWT middleware to authorize the request by ensuring the access\_token is valid.

This is assuming that the access\_token is a JWT, since OpenIddict, for example, can generate access tokens as either a JWT or it's own proprietary format - in which case you need to use their OAuth introspection middleware instead of the JWT middleware

* [IdentityServer4](https://github.com/IdentityServer/IdentityServer4):

an OpenID Connect and OAuth 2.0 framework for ASP.NET Core 2.

* [AspNet.Security.OpenIdConnect.Server](https://github.com/aspnet-contrib/AspNet.Security.OpenIdConnect.Server):

is an advanced OAuth2/OpenID Connect server framework for both ASP.NET Core 1.x/2.x and OWIN/Katana 3.x/4.x, designed to offer a low-level, protocol-first approach.

* [OpenIddict](https://github.com/openiddict/openiddict-core):

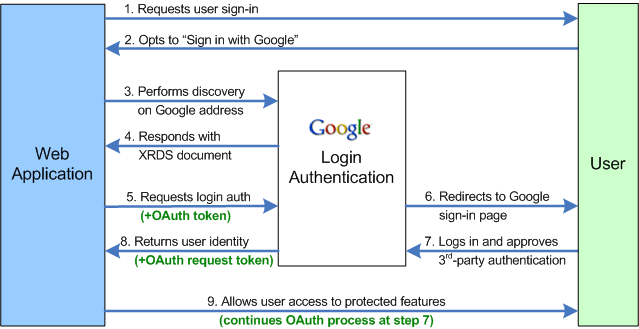
OpenIddict aims at providing a simple and easy-to-use solution to implement an OpenID Connect server in any ASP.NET Core 1.x or 2.x application.

OpenIddict is based on AspNet.Security.OpenIdConnect.Server to control the OpenID Connect authentication flow and can be used with any membership stack, including ASP.NET Core Identity.

Also have checked that all of them use well the [ASP.NET Core Identity](https://github.com/aspnet/Identity) as a membership system.

And so my current understanding is that IdentityServer4 and OpenIdConnect.Server are two alternative frameworks that solve the same problem. The main difference is the list of supported ASP.NET Core versions.

Regarding Openiddict - it is a kind of extension to simplify server creation based on AspNet.Security.OpenIdConnect.Server.



**ID Tokens vs Access Tokens**. The **ID Token** is a security **token** granted by the OpenID Provider that contains information about an End-User. ... **Access tokens**, on the other hand, are not intended to carry information about the user. They simply allow **access** to certain defined server resources