Securing microservices

(Demo prep)

- Remove all tokens from Postman
- Clear Authorization values in Postman

Standing up Identity Server 4

It's time to turn our Identity microservice into an OAuth 2.0 authorization server and an OpenID Provider. Let's start by integrating IdentityServer into our service.

In Identity Microservice

1. Add IdentityServer packages:
   dotnet add package **IdentityServer4**
   dotnet add package **IdentityServer4.AspNetIdentity**

2. Add **Settings** directory

3. Add **IdentityServerSettings.cs**:

```
namespace Play.Identity.Service.Settings
{
  public class IdentityServerSettings
  {
    public IReadOnlyCollection<ApiScope> ApiScopes { get; init; } = Array.Empty<ApiScope>();
    public IReadOnlyCollection<Client> Clients { get; init; } = Array.Empty<Client>();
  }
}
```

4. Update **Startup.cs (collapse explorer)**

```
public void ConfigureServices(IServiceCollection services)
{
  …
  Configuration.GetSection(nameof(MongoDbSettings)).Get<MongoDbSettings>();
  var identityServerSettings = new IdentityServerSettings();

  services.AddDefaultIdentity<ApplicationUser>()
    .AddRoles<ApplicationRole>()
    .AddMongoDbStores<ApplicationUser, ApplicationRole, Guid>
    (
      mongoDbSettings.ConnectionString,
      serviceSettings.ServiceName
    );
```

```
    services.AddIdentityServer()
        .AddInMemoryApiScopes(identityServerSettings.ApiScopes)
        .AddInMemoryClients(identityServerSettings.Clients);

    services.AddControllers();

    services.AddSwaggerGen(c =>
    {
        c.SwaggerDoc("v1", new OpenApiInfo { Title = "Play.Identity.Service", Version = "v1" });
    });
}

public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{
    …
    app.UseRouting();

    app.UseIdentityServer();
    app.UseAuthorization();
    …
}
```

5. Show OpenId discovery document:

https://localhost:5003/.well-known/openid-configuration

"The discovery document is a standard endpoint in identity servers. The discovery document will be used by your clients and APIs to download the necessary configuration data."

6. Add **IdentityResources** to **IdentityServerSettings**:

```
public IReadOnlyCollection<IdentityResource> IdentityResources =>
    new IdentityResource[]
    {
        new IdentityResources.OpenId()
    };
```

7. Update **Startup.cs**:

```
services.AddIdentityServer()
        .AddAspNetIdentity<ApplicationUser>()
        .AddInMemoryApiScopes(identityServerSettings.ApiScopes)
        .AddInMemoryClients(identityServerSettings.Clients)
```

.AddInMemoryIdentityResources(identityServerSettings.IdentityResources)
.AddDeveloperSigningCredential();

8. Start server and refresh OpenId discovery document

9. Show **scopes_supported** and **claims_supported** in OpenId discovery document

10. Show the generated **tempkey.jwk**

"IdentityServer needs some signing key material to sign tokens. At first startup, IdentityServer will create a developer signing key for you, it's a file called tempkey.jwk."

In the next lesson we will enable our Postman app to verify the identity of the logged in user via OpenID Connect.