**Laboratorio Azure Identity (90 minutos)**

**Práctica: Registro de una Aplicación en Microsoft Identity Platform**

**Objetivo:** Los alumnos aprenderán a registrar una aplicación en Microsoft Identity Platform, obtener las credenciales necesarias y comprender los permisos requeridos para la autenticación y autorización.

**Requisitos Previos:**

* Cuenta de Microsoft (puede ser personal o de una organización)
* Acceso al portal de Azure ([**https://portal.azure.com**](https://portal.azure.com))
* Conocimientos básicos de autenticación y autorización

**Pasos para Registrar una Aplicación**

**Paso 1: Ingresar al Portal de Azure**

1. Ir a [Azure Portal](https://portal.azure.com).
2. Iniciar sesión con una cuenta de Microsoft.

**Paso 2: Crear un Registro de Aplicación**

1. En la barra de búsqueda, escribir "Azure Active Directory" y seleccionarlo.
2. En el panel izquierdo, hacer clic en "**Registros de aplicaciones**".
3. Hacer clic en **"Nuevo registro"**.
4. Completar los siguientes campos:
   * **Nombre**: Elegir un nombre representativo para la aplicación.
   * **Tipos de cuenta compatibles**: Elegir según sea necesario:
     + Solo cuentas en este directorio de Azure AD
     + Cuentas en cualquier directorio de Azure AD
     + Cuentas de Azure AD y Microsoft (cuentas personales)
   * **URI de redirección** (opcional en este paso).
5. Hacer clic en **"Registrar"**.

**Paso 3: Obtener las Credenciales de la Aplicación**

1. Después del registro, anotar los siguientes valores:
   * **ID de la aplicación (cliente)**
   * **ID del directorio (tenant)**
2. En la sección "Certificados y secretos", generar un nuevo secreto de cliente:
   * Hacer clic en "Nuevo secreto de cliente".
   * Elegir una duración y asignar un nombre.
   * Copiar el valor generado (no se podrá ver nuevamente).

**Paso 4: Configurar los Permisos de la API**

1. En el menú izquierdo, ir a "Permisos de API".
2. Hacer clic en "Agregar un permiso".
3. Elegir "APIs de Microsoft" y seleccionar **Microsoft Graph**.
4. Elegir los permisos según la aplicación (delegados o de aplicación).
5. Si se requieren permisos de administrador, hacer clic en "Conceder consentimiento de administrador".

**Paso 5: Configurar la URI de Redirección (Opcional)**

1. Ir a "Autenticación" en la configuración de la aplicación.
2. Agregar una **URI de redirección** si se desea autenticar usuarios (ejemplo: https://localhost:5001/signin-oidc).
3. Guardar los cambios.

**Ejercicio Final**

Cada alumno deberá:

* Registrar una aplicación siguiendo los pasos anteriores.
* Compartir el **ID de la aplicación (cliente)** con el instructor: Cada alumno debe compartir:
  + **ID de la aplicación (cliente)**
  + **ID del directorio (tenant)**

Esto se encuentra en la página de resumen de la aplicación en **Azure AD**.

* Generar un secreto de cliente y demostrar cómo lo utilizaría en una aplicación:

Para obtener un secreto de cliente:

1. En el menú lateral, ir a **"Certificados y secretos"**.
2. Hacer clic en **"Nuevo secreto de cliente"**.
3. Asignar un nombre y seleccionar la duración.
4. Copiar y almacenar el valor del secreto de cliente (ya que no podrá verse nuevamente).

Demostrar el Uso del Secreto en una **Aplicación ASP.NET Core**

Cada alumno deberá integrar la autenticación en una aplicación ASP.NET Core con Azure AD.

4.1 Configurar appsettings.json

Los alumnos deben agregar las credenciales en su archivo de configuración:

{

"AzureAd": {

"Instance": "https://login.microsoftonline.com/",

"TenantId": "your-tenant-id",

"ClientId": "your-client-id",

"ClientSecret": "your-client-secret",

"CallbackPath": "/signin-oidc"

}

}

4.2 Implementar la Configuración en Program.cs

var builder = WebApplication.CreateBuilder(args);

builder.Services.AddAuthentication(OpenIdConnectDefaults.AuthenticationScheme)

.AddMicrosoftIdentityWebApp(builder.Configuration.GetSection("AzureAd"));

builder.Services.AddControllersWithViews();

builder.Services.AddAuthorization();

var app = builder.Build();

app.UseAuthentication();

app.UseAuthorization();

app.MapControllerRoute(

name: "default",

pattern: "{controller=Home}/{action=Index}/{id?}");

app.Run();

4.3 Incluir Login y Logout en AccountController.cs

using Microsoft.AspNetCore.Authentication;

using Microsoft.AspNetCore.Mvc;

public class AccountController : Controller

{

public IActionResult Login()

{

return Challenge(new AuthenticationProperties { RedirectUri = "/" }, "OpenIdConnect");

}

public IActionResult Logout()

{

return SignOut(new AuthenticationProperties { RedirectUri = "/" }, "Cookies", "OpenIdConnect");

}

}