using Microsoft.IdentityModel.Clients.ActiveDirectory;

using System;

using System.Net.Http;

using System.Net.Http.Headers;

using System.Threading.Tasks;

namespace ComplianceAPI

{

class Program

{

static void Main(string[] args)

{

var tenantId = "896d9569-0103-45a4-be80-abcdefecacfb";

var clientId = "c95d36dc-9a41-478f-se12-638375414a1a";

var secret = "6edEKcNhVN0jFLGSWTqr5XZGWSOzKJ4cLNw8hETSIs=";

var messageHandler = new OAuthMessageHandler(tenantId, clientId, secret, new HttpClientHandler());

using (HttpClient httpClient = new HttpClient(messageHandler))

{

httpClient.BaseAddress = new Uri("https://manage.office.com");

httpClient.Timeout = new TimeSpan(0, 2, 0); //2 minutes

string endpoint = $"/api/v1.0/tenantId/activity/feed/subscriptions/start?contentType=Audit.Sharepoint";

HttpRequestMessage message = new HttpRequestMessage(new HttpMethod("POST"), endpoint);

var resp = httpClient.SendAsync(message, HttpCompletionOption.ResponseHeadersRead).Result;

var response = resp.EnsureSuccessStatusCode();

HttpContent httpContent = resp.Content;

Console.WriteLine("Processing completed");

}

}

/// <summary>

///Custom HTTP message handler that uses OAuth authentication thru ADAL.

/// </summary>

class OAuthMessageHandler : DelegatingHandler

{

private AuthenticationHeaderValue authHeader;

public OAuthMessageHandler(string tenantId, string clientId, string secret, HttpMessageHandler innerHandler)

: base(innerHandler)

{

var authUrl = $"https://login.microsoftonline.com/{tenantId}/oauth2/token";

AuthenticationContext authenticationContext = new AuthenticationContext(authUrl, false);

var clientCredential = new ClientCredential(clientId, secret);

AuthenticationResult authResult = authenticationContext.AcquireTokenAsync("https://manage.office.com", clientCredential).Result;

authHeader = new AuthenticationHeaderValue("Bearer", authResult.AccessToken);

}

protected override Task<HttpResponseMessage> SendAsync(

HttpRequestMessage request, System.Threading.CancellationToken cancellationToken)

{

request.Headers.Authorization = authHeader;

return base.SendAsync(request, cancellationToken);

}

}

}

}