Process to retrieve data from WebAPI

The solution describes in phases the use of PowerShell and T-SQL Scripts to perform extract, transform and load operations to a SQL Server database via a GET REST API request.

# Retrieving and Transforming data via a GET REQUEST

The PowerShell script below retrieves the latest exchange rate data using the GET Method. The API content type was JSON and was transformed and saved as a CSV file.

$Params = @{

    "Uri" = 'https://currency-conversion-and-exchange-rates.p.rapidapi.com/latest?from=USD&to=EUR%2CGBP'

    "Method" = 'Get'

    "Headers" = @{

        "ContentType" = 'application/json'

        'Authorization' = 'Bearer <AccessToken>'

        "X-RapidAPI-Key" = 'c892889770msh34b4611091f1e65p1b3a02jsnaca2efec6843'

        "X-RapidAPI-Host" = 'currency-conversion-and-exchange-rates.p.rapidapi.com'

    }

}

$response = Invoke-RestMethod @Params

$response |

    Select-Object timestamp, base, success, date, `

        @{Name = "USD"; Expression = {$\_.rates.USD}}, `

        @{Name = "GBP"; Expression = {$\_.rates.GBP}}, `

        @{Name = "NGN"; Expression = {$\_.rates.NGN}}, `

        @{Name = "EUR"; Expression = {$\_.rates.EUR}} |

        ConvertTo-Csv -NoTypeInformation | Out-File D:\NLNG\Work\Csv\ExchangeRates.csv

# Updating the database via CRUD

The CSV file egressed via the API as a CSV would be inserted into a SQL Server database using a BULK INSERT Script below:

--Add Exchange Rates Data

BULK INSERT [dbo].[ExchangeRates] FROM 'D:\NLNG\Work\Csv\ExchangeRates.csv' WITH

(

    DATAFILETYPE = 'widechar'

    ,FIRSTROW = 2

    ,FIELDTERMINATOR = ','

    ,ROWTERMINATOR = '\n'

    ,TABLOCK

    ,FORMAT = 'CSV'

    ,FIELDQUOTE = '"'

);

GO

The T-SQL script would be automated to run on schedule using a CRON Job and PowerShell Script below:

Write-Host -foreground green 'Inserting Exchange Rates Data...'

Invoke-SqlCmd -Server localhost -Database "NLNGProjects" -InputFile "D:\NLNG\Work\Script\UpdateDatabase-1.sql"

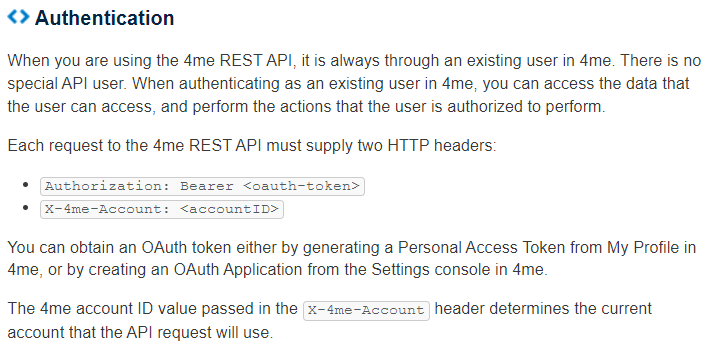
Write-Host -foreground green 'Discarding CSV file...'

# $z = $host.UI.RawUI.ReadKey("NoEcho,IncludeKeyDown")

Remove-Item -Path D:\NLNG\Work\Csv\ExchangeRates.csv

# 4Me API Requirements

With reference to the 4me API Documentation, OAuth authentication access token is required from the API Provider to connect to the API URI: <https://api.4me.com/v1/export>



An attempt to connect to the REST API without an access token or credentials returns a **401 Unauthorized response status code**, indicating lack of valid authentication.

try {

    $Params = @{

        "URI" = 'https://api.4me.com/v1/export'

        "Method" = 'GET'

        "Headers" = @{

            "Content-Type" = 'application/json'

            "Authorization" = 'Bearer'

        }

    }

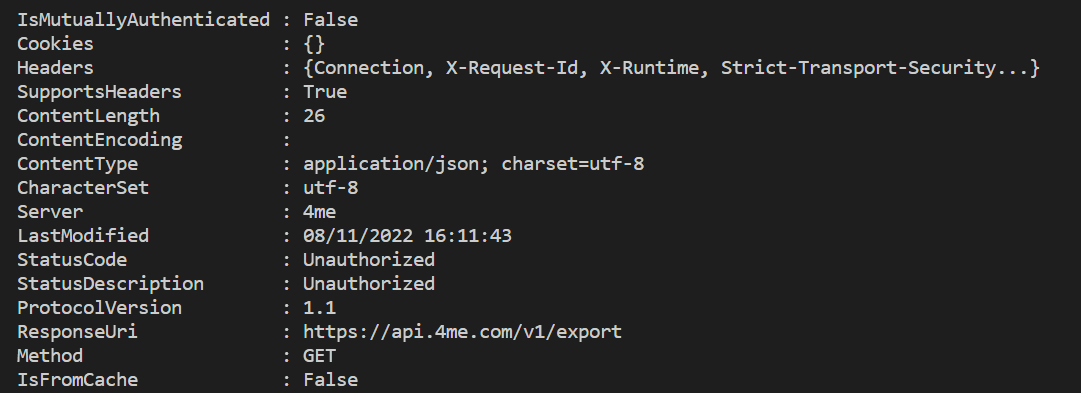
    Invoke-RestMethod @Params

}

catch {

    $\_.Exception.Response

}



# OAuth Authentication using Microsoft Graph API

The Microsoft Graph API is allows connecting to data in the Microsoft 365 environment and **requires the OAuth authentication method.** Reference is made to this to illustrate how a connection can be made to an API of OAuth authentication requirements.

The PowerShell Script below propagates a HTTP request via Microsoft Graph API to my Sandbox account to retrieve data regarding my Microsoft 365 developer account.

$AccessToken = ConvertTo-SecureString 'eyJ0eXAiOiJKV...' -AsPlainText -Force

$Params = @{

    "URI" = 'https://graph.microsoft.com/v1.0/me'

    "Method" = 'GET'

    "Authentication" = 'OAuth'

    "Token" = $AccessToken

    "ContentType" = 'application/json'

}

$Response = Invoke-RestMethod @Params

$Response |

    Select-Object displayName, @{Name="Phone";Expression={$\_.businessPhones[0]}}, mail | ConvertTo-Json

The retrieved data is shown below as a JSON document:

{

    "displayName": "Richard Ogoma",

    "Phone": "23408183064412",

    "mail": "richard@7ykj10.onmicrosoft.com"

  }