**DynamicETL\_Dashboard:**

* **Description:**
  + Scripts and object framework useful in automating key steps of the ETL generation, debugging and validation process.
* **Scripts used in application:** 
  + **GenerateNewETL.py:**

1. Reads one or more files in folder matching regular expression containing data that you wish to implement in a new ETL.
2. Generates column report detailing appropriate T-SQL types, attributes, relationships between columns. If multiple files were read then details how columns have changed over time.
3. Generate .sql table definition based on column types using predetermined template (currently for use in MetricsDYETL).
4. Append new ETL to DynamicETL.Service appsettings-template.json file with default attributes.
   * **TestETLPipeline.py:** 
     + If testing in LOCAL:
       1. Post arguments using postargs.json to local instance of DynamicETL.WebAPI.
       2. Run DynamicETL.Service to push data into target table.
       3. Query ETL’s T-SQL table for inserted data and generate report comparing inserted data with data in source file, detailing specific differences between rows.
     + If testing in QA/UAT/STG:
       1. Using file configured in DynamicETL.WebApi postargs.json, drop file to ETL’s configured FileWatcher location.
       2. Wait for file to be sucked up by ETL.
       3. Query ETL’s T-SQL table for inserted data and generate report comparing inserted data with data in source file, detailing specific differences between rows.
   * Every script has .json file with same name as script (ex: GenerateNewETL.json, TestETLPipeline.json).

* **Software Requirements:**
  + Python >= 3.6
  + Packages listed in requirements.txt.
* **Command line usage:**
  + “python <PathToScript.py>”

**Script Arguments:**

* **GenerateNewETL.py** (GenerateColumnAttributesReport.json)**:**
  + Required Arguments:
    - **“data”:** Dictionary with parameters “path” with path to folder containing one or more files (string), and optional “delim” denoting which delimiter to use to read file if csv (string).
    - **“etlname”:** Name you want to use for the new ETL (string).
    - **“outputfolder”:** Path tofolder you want to output table definition, report and updated DynamicETL.Service appsettings.json and appsettings-template.json (string, must exist).
    - **“filedatereg”**: Dictionary with parameters “Regex” to store regular expression to use for pulling file date from each file’s name, and “DateFormat” for
    - **“filenamereg”:** Regularexpression for
    - **“tablename”:** Nameof T-SQL table to use with ETL. Will be used in the generated table definition (string).
  + Optional Arguments:
    - **“allnull”**: Put true if want to set all arguments to NULL. Ideal if possible that columns in dataset will be nullable in future (boolean).
* **TestETLPipeline.py** (in TestETLPipeline.json)**:**
  + Required Arguments:
    - **“etlname”:** Name of ETL. Must be configured in DynamicETL.Service appsettings.json located in ETLDashboard.json:: in QA/UAT/STG and in local copy (string).
    - **“filedate”**: Date of file used to test pipeline (date string).
    - **“reportpath”:** Path to output report comparing input data and output data (string, must be .xlsx).
    - **“postargspath”:** Path to postargs.json file containing REST arguments to be POSTed to DynamicETL.WebAPI. Required whether testing locally or QA/UAT/STG since the script will read from (string, must be json).
    - **“testmode”:** One of “LOCAL”/”QA”/”UAT”/”STG” (case insensitive) denoting which version of the ETL you are testing (string).
  + Optional Arguments:
    - **“pkey”:** Specific columns you want to use to compare two rows, i.e. the primary key of the dataset (list ofstrings or string).
    - **“ignorecols”:** Columns you want to ignore when comparing rows. Use if the ETL transforms the dataset in some way. FileDate and RunDate columns are always ignored (list of strings, string)
    - **“comparefile”:** Path to file containing data you want to compare to data loaded into ETL table, in case you do not want to use the file listed in postargs.json.
* **Universal arguments** (in ETLDashboard.json): Arguments that are used in both GenerateNewETL and TestETLPipeline:
  + **“dynamicetlservicepath”**: Path to local DynamicETL.Service executable for testing the ETL pipeline locally (i.e. running the TestETLPipeline.py with “testmode” : “LOCAL”).
  + **“chromedriverpath”**: Path to chromedriver.exe (string regex pattern).
  + **“config”**: Path to config.json used to fill in environment variables in json files (string, must be json).
  + **“etlfilepaths”:** Path tojson file containing filepaths used by ETL to pick up data files (string, must be json).
  + **“filewatcherappsettingstemplatepath”:** Path to json file containing FileWatcher appsettings.json (string, must be json).
  + **“filetransferurl”:** Path to url used by filetransfer service (string).
  + **“logpath”:** Path to logfiles generated by applications like DynamicETL.Service (string, must be json).
  + **“serviceappsettingspath”:** Path tojson file containing DynamicETL.Service appsettings contents (string, must be json).
  + **“serviceappsettingstemplatepath” :** Path to json file containing DynamicETL.Service appsettings-template contents (string, must be json).
  + **“webapiurl” :** URL used by local instance of DynamicETL.WebAPI (string).
  + **“webapipath” :** Path to local DynamicETL.WebAPI .dll (string, must be dll).
* **Supported environment variables:** that can be filled in by any used script argument .json file. Additions can be made by adding to the FillUniversalEnvironmentVariables() function in Utilities\Helpers.py.
  + **“{LocalPath}”:** String fill be filled in with path local to DynamicETL\_Dashboard.py.