Overview

In previously build of QA Automation script in https://github.com/taigers/QA-Automation, after document(s) uploaded to iMatch, a CSV file would be generated which contain the data points differences between given Gold data and extracted data from uploaded document(s). However, this generated file can't be directly used for QA accuracy report.

In MOM project, further improvements has been made for this script so that after document(s) uploaded, an accuracy report will also generated that can be used directly by QAs. The main issue of this script is it's written in Java while the script developed in Taiger's Github is fully written in Python. Therefore, this documentation will explain the improvements and changes made from the previous QA Automation but still using Python as the script language.

What does this script do?

- Calls Extract's (iMatch's) APIs to upload files in bulk (without crashing pipeline).
- Get HTML for GATE's GUI.
- Get extracted data for each documents uploaded.

What can we use the script for?

- Bulk upload of documents for any given type and sub-type.
- Get the HTML outputs of OCR for the uploaded documents (Can be used for GATE Development).
- Get extracted data then generate an excel sheet that can be used for accuracy report.

What files can this script process ?

Any formats supported by iMatch can be processed by iMatch, e.g. .png, .pdf, .jpg, .tif. Other formats can be added in 'accepted_formats' in **config.py.**

Limitations of this script

- Fail for HTTPS authentication.
- Can upload one document type at once.
- Will not processed the document if its name is already recorded at iMatch.

Technical Documentation

Prerequisites

- Python
- PyCharm or any other IDE

Libraries needed

- pandas
- requests
- xlsxwriter
- certifi
- chardet
- Click
- Flask
- idna
- itsdangerous
- Jinja2
- Markupsafe
- numpy
- python-dateutil
- pytz
- six
- urrlib3
- Werkzeug

How to use the script

- Clone QA Automation repository from Github to local machine using this command
  git clone https://github.com/taigers/QA-Automation
- Create virtual environment for this script.
- Copy/cut QA Automation repository to previously created virtual environment.
- Install all dependencies required within requirements.txt within the virtual environment
- Made changes to config.py
- Run the processfolder.py

Details on config.py

URL

The **url** variable can be filled with the endpoint of targeted iMatch service that will be uploaded by documents given. The **url** variable also related to **host** variable because the **host** variable will be used in **processFolder.py** to get the token_id from given endpoint. If possible, include the gateway of the endpoint within the **url** variable.

Example:
Include gateway within the url variable

```
# --- URL ---
# IMDA
url = 'https://extract-dev.taiger.io'

# Sompo
# url = 'http://imatch-macquarie1.taiger.io:8080'
# url = 'http://168.63.244.231:8080'

# Do not change unless the iMatch is not a kubernetes deployment.
# Kubernetes deployment needs all the calls to be directed to /gateway instead of :8080
# host = url + 'gateway'
# For normal deployments;
host = url
```

Gateway included in URL

Gateway combined in the host variable

```
# --- URL ---
# IMDA
url = 'https://extract-dev.taiger.io'

# Sompo
# url = 'http://imatch-macquarie1.taiger.io'
# url = 'http://168.63.244.231'

# Do not change unless the iMatch is not a kubernetes deployment.
# Kubernetes deployment needs all the calls to be directed to /gateway instead of :8080
# host = url + 'gateway'
# For normal deployments;
host = url + ':8080'
```

The list of endpoints that can be used for specific projects.

Project	url
IMDA	https://extract-dev.taiger.io
Medical/Sompo	http://imatch-macquarie1.taiger.io:8080 http://168.63.244.231:8080

Credentials

For variables in credentials category, there are two variables that will be used. First one is username and second one is password. These credentials can be obtained by asking developers.

Example:

```
# --- Credentials ---
username = 'lilik'
# password = 'Password123@' # Sompo
password = 'Password@123' # IMDA
```

Document Type and Subtype

For a particular project, there are several document type and subtype that can be used to upload certain document related to the subtype. Here's the example list of the docType and subType variables that can be used for a project.

Project	docType	subType
IMDA	IMDA	AttendanceSheet SkillFuture FinalCert CertificateOfCompletion DCA
Medical/Sompo	Medical	medicalCertificates medicalInvoices medicalCertificates_Sompo medicalInvoices_Sompo
VFS	Identity	passport

Example:

```
# --- docType and subtype ---
docType = 'IMDA'
subType = 'SkillFuture'
```

Accepted Formats

The **accepted_formats** variable is a list of formats that can be used and accepted by iMatch. The default accepted formats are **png**, **pdf**, **jpg**, and **tif**. However if a project supports other formats e.g. **xls**, **xlsx**, **doc**, or **docx**, it can be added to this variable.

```
# --- Accepted Formats ---
accepted_formats = ['png', 'pdf', 'jpg', 'tif']
```

Folder Path

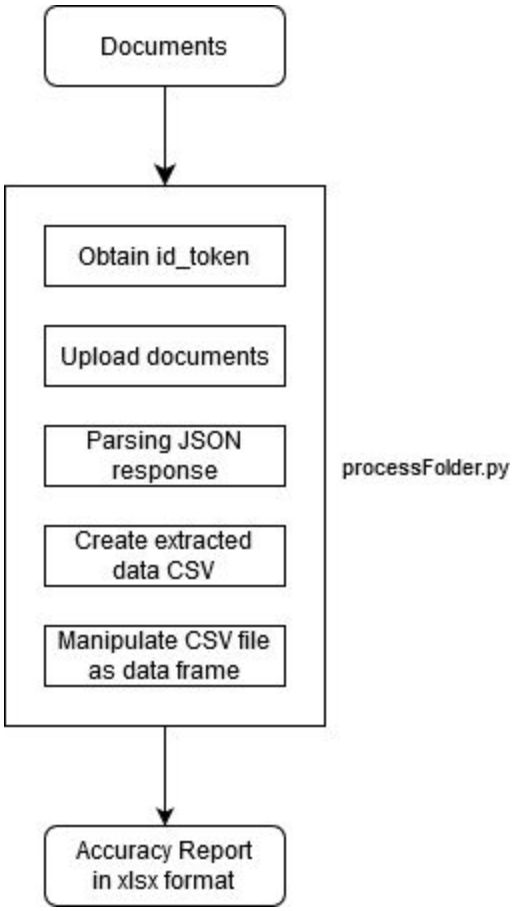
Any documents that will be uploaded must be put within **test** folder. Inside the script, there's Gold Data Path but this path variable won't be used anymore because the purpose of this data is to be compared with data extracted from iMatch. One improvement from the new script is the output result only show the extracted data and the gold data still filled manually by QA.

```
# Folder path and Gold csv path
folder_to_process = os.getcwd() + "/test/"

# --- Gold Data Path
# gold_csv_path = folder_to_process + "GOLD.csv"
```

Details on processFolder.py

Flowchart



Functions

Comments on how each function works have been put within the script.