**1. Name of program:**

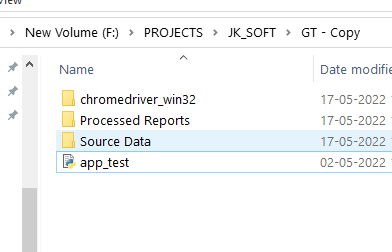
Operational Process Automation – Reporting.

**2. Objective:**

A code based solution using Python and SQL that will automate the process of importing data, preparing reports, and delivering reports on/at scheduled days/times, to designated stakeholders.

**3. Scope – Summary:**

* + Project - Requirements
  + Folder Structure Creation for the program flow
  + Source file creation through web-scraping
  + Program-Execution:
    - Read the source files for data
    - Export the read data to DBMS as per the provided mappings.
    - Report generation from DBMS to an new excel file with necessary customizations.
  + **Project – Requirements**
  + Pyodbc – To establish connection between Python code and DBMS
  + Pandas – To retrieve and manipulate source file data
  + Chardet – for encoding
  + Xlsxwriter – to write data from data frames to external excel files
  + Glob – tp search for files in a respective path
  + Openpyxl – for excel formatting
  + Email – to enabling E-mailing functionality from script
  + Fernet – for data encryption
  + **Folder Structure for the program flow:**
  + The following folders will be manually created at the path where the code will be placed for the ease of flow of program
  + Processed Reports
  + Source Data
  + chromedriver\_win32



* Processed Reports folder - The path that will hold the processed output files
* Source data folder - The path that will hold the Source files
* chromedriver\_win32 - The path that will hold the chrome exe for web scraping purpose.
* Source file creation through web-scraping (chrome.exe should be replaced as per the machine’s OS of the user)
* **Program-Execution:**
* Reads the source files for data (Beneficiary File):
* Thus the acquired source files namely Reports Automation - Beneficiary Data (HQ Based) (for ImmiLytics), Reports Automation - Case Data (HQ Based) will be renamed as per the below conventions
* Reports Automation\_Beneficiary Data\_(client-name) \_mmddyyyy
* Reports Automation\_Case Data\_(client-name)\_mmddyyyy

and will be placed in the source file folder.

* Now the code searches and picks for the Beneficiary data file in the source folder contains the current date in its filename.
* This uses pandas framework to read the content of the file and stores within as data frames
* Now the code reads the number of rows the dataframe has and loops repetitive for ‘n’ number of rows the the dataframe holds.
* During each looping it reads the data of all the columns of the each respective rows and assigns to the respective columns as defined by the mappings and pushes/adds the data of each rows to the Beneficiary table as denoted in the respective SQL code.
* The loop ends once feeding data of all the rows in dataframe that’s extracted from the source beneficiary file.
* **Reads the source files for data (Case data File):**
* Once the insertion of data to the DBMS from the beneficiary data is completed the same identical process is restarted for case file as elaborated below
* The code now searches and picks for the Case data file in the source folder that contains the current date in its filename
* This uses pandas framework to read the content of the file and stores within as data frames.
* Now the code reads the number of rows the dataframe has and loops repetitive for ‘n’ number of rows the the dataframe holds.
* During each looping it reads the data of all the columns of the each respective rows and assigns to the respective columns as defined by the mappings and pushes/adds the data of each rows to Case table as denoted in the respective SQL code.
* The loop ends once feeding data of all the rows in dataframe that’s extracted from the source beneficiary file.
* **Report generation:**
* Now once all the datas from the beneficiary files are available in the DBMS it’s time to query all the tables as per the requirements to create a output file as required.
* The code queries the Client data table in the database for record that has a definite delivery date in it.
* And creates the Output file with the following tabs:
* Open Cases -
* Filed & Pending Cases
* Approved & Closed Cases
* Cases on Hold
* Priority Date Report
* Active Beneficiary List
* Further it moves to create Document Expiration Report with the below tab name
* Doc Exp Report - 8 Months Out
* The sheet enlists all active beneficiaries whose any one of the below criteria did not yet expire
* I797ExpirationDate
* FinalNivDate
* VisaPedDate
* EadExpirationDate
* AdvanceParoleExpirationDate
* Ds2019ExpirationDate
* ReEntryPermitExpirationDate
* GreenCardExpirationDate
* MostRecentPassportExpirationDate
* VisaExpirationDate BETWEEN
* **Sending mail:**
* Once all the reports are ready it’s time to mail the clients
* The code checks for availability of the recipient mail id’s from ClientsDetails table further checks if the SEND\_MAIL option is enabled/disabled
* In case enabled:
* The code gets the mail ID’s of the primary recipient and secondary recipient
* Further only in an account of SEND\_MAIL\_TO\_CLIENT is too enabled the secondary recipients are also considered.
* An send\_exp\_report option is set to provide an option to whether include are not the “Document Expiration Report” in the mailing attachments.
* Once the report being sent the “report\_sent\_on” column of clientsdetails table are updated accordingly.

-------------------------------------------------------------------------------

‘