UTM

UNIVERSITI TEKNOLOGI MALAYSIA

SEMESTER 1 2022/2023

SMJE 4383

ADVANCED PROGRAMMING

ASSIGNMENT 1

Automate the CSV Generation Process using

Robotic Process Automation (RPA) & Python

NAME	Matric No.	Signature
1. LI DAHUA	A19MJ5039	li
2. Tew Jia Jian	A19MJ0110	
Name of Lecturer		**SECTION**
Assoc Prof Ir Dr Zool Hilmi Ismail		01

1.0 INTRODUCTION

1.1 Problem Background

The RPA robot process automation can replace or assist humans in completing repetitive work and tasks in digital devices such as computers and mobile phones. Python language is easy to use and a powerful tool to help the robot process automation tool for the CSV generation process. It can complete repetitive and monotonous process work, reduce manual errors, improve operational efficiency, and reduce operating costs in the CSV document process.

CSV files are widely used for data exchange and storage. The process of generating a CSV file typically involves collecting data from various sources, cleaning and transforming the data, and writing the data into a CSV file. This process can be time-consuming and easily has errors, especially it is done manually.

CSV Buddy is a simple, user-friendly RPA software that can be used to create, edit, and view CSV files. It can be used to automate the data collection and retrieval process, while Python can be used to process the data and write it into a CSV file. By using both tools, we can achieve efficient, reliable, and accurate CSV generation process. This will also bring several benefits, such as reduced manual errors, increased efficiency and speed, improved data quality, and cost savings.

1.2. Problem Statement

The generation of CSV files for data exchange and storage processes is required by many organizations. But the manual process of collecting data, cleaning and transforming it, and writing it to a CSV file is time-consuming, prone to errors, and can be a bottleneck for data management processes.

Therefore, this project can improve the accuracy and efficiency of the process, by automating the data processing, and writing to a CSV file using Robotic Process Automation (RPA) tools and Python. The solution should reduce manual errors, increase speed and efficiency, improve data quality, and reduce the cost of the organization.

1.4 Project Objective

- To use software CSV Buddy to create, edit, and view CSV files, and simplify the data entry process.

- Use RPA tools & Python code to automatically copy data from another CSV file into a new CSV in this software.

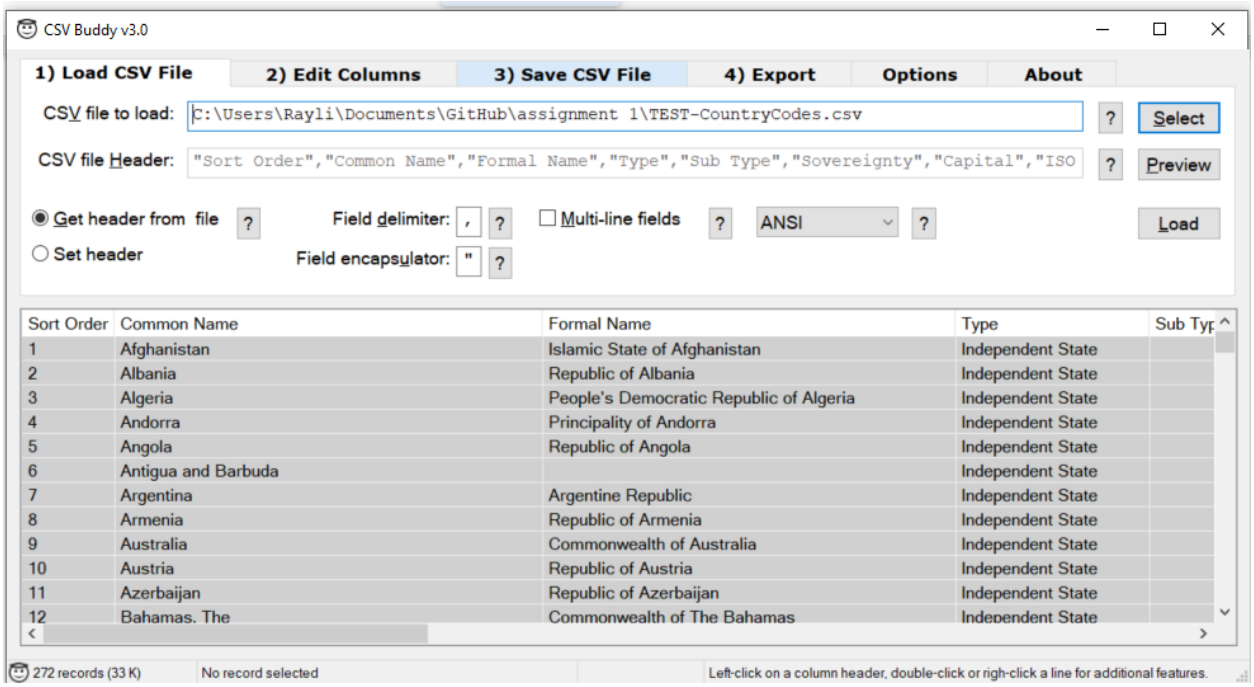
- To automate the data collection process for a more efficient and streamlined CSV generation process.

2.0 METHODOLOGY

CSV Buddy is designed to be an easy-to-use and efficient tool for working with CSV files, making it a useful tool for organizations and individuals who frequently work with data in CSV format. It provides features for creating, editing, and viewing CSV files, making it a useful tool for managing and manipulating data.

CSV Buddy provides an interface for manually entering data into a CSV file, or editing existing data. It also allows users to visualize data in a table format, making it easier to understand and analyze.

CSV Buddy supports the import and export of data from and to other file formats, such as HTML or XML. It can use the "Import" feature to import data from another CSV file or a spreadsheet.



CSV GUI

We use this CSV buddy import a CSV file of country code and edit an CSV file for python to process.

Python Scripts

This python script use to automates the process of copying data from one CSV file to another and sending a notification email.

```python
import csv
import smtplib

# specify the source and destination CSV files
source_file = "TEST-CountryCodes.csv"
destination_file = "Testing.csv"

# open the source file and read its data
with open(source_file, "r") as src:
    reader = csv.reader(src)
    data = [row for row in reader]

# open the destination file and write the data from the source file
with open(destination_file, "w", newline='') as dest:
    writer = csv.writer(dest)
    writer.writerows(data)

# send a notification email
server = smtplib.SMTP("smtp.gmail.com",587)
server.starttls()
server.login("lidahua@graduate.utm.my", "Password")

from_email = "lidahua@graduate.utm.my"
to_email = "raylidahua@gmail.com"
subject = "CSV Data Copied"
body = "The data from the source CSV file has been successfully copied to the destination CSV file."

message = f"Subject: {subject}\n\n{body}"

server.sendmail(from_email, to_email, message)
server.quit()
```

Scripts

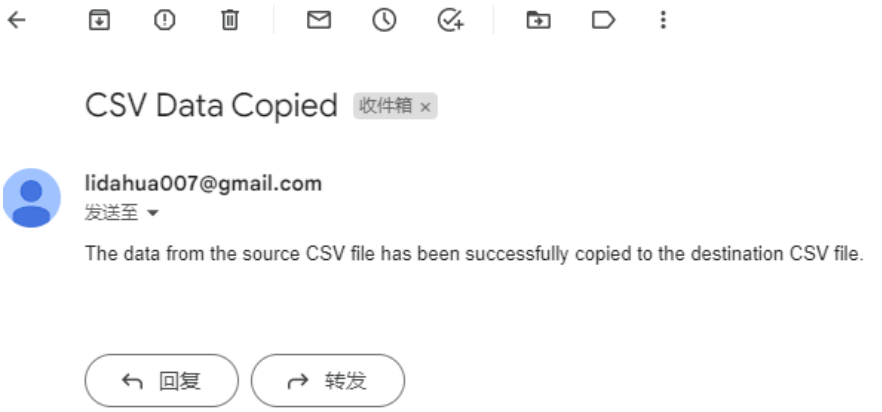
The first step is to import the needed libraries. The csv library is used for reading and writing CSV files, and the smtplib libraries are used for sending the notification email.

The source Country Code CSV file is opened using the with statement, which ensures that the file is automatically closed when the code block is exited. The data from the source file is read row by row using the csv.reader attribute and stored in a list called data. Then the target CSV file is opened using the with statement, and the data from the data list is written to the file using the csv.writer attribute.

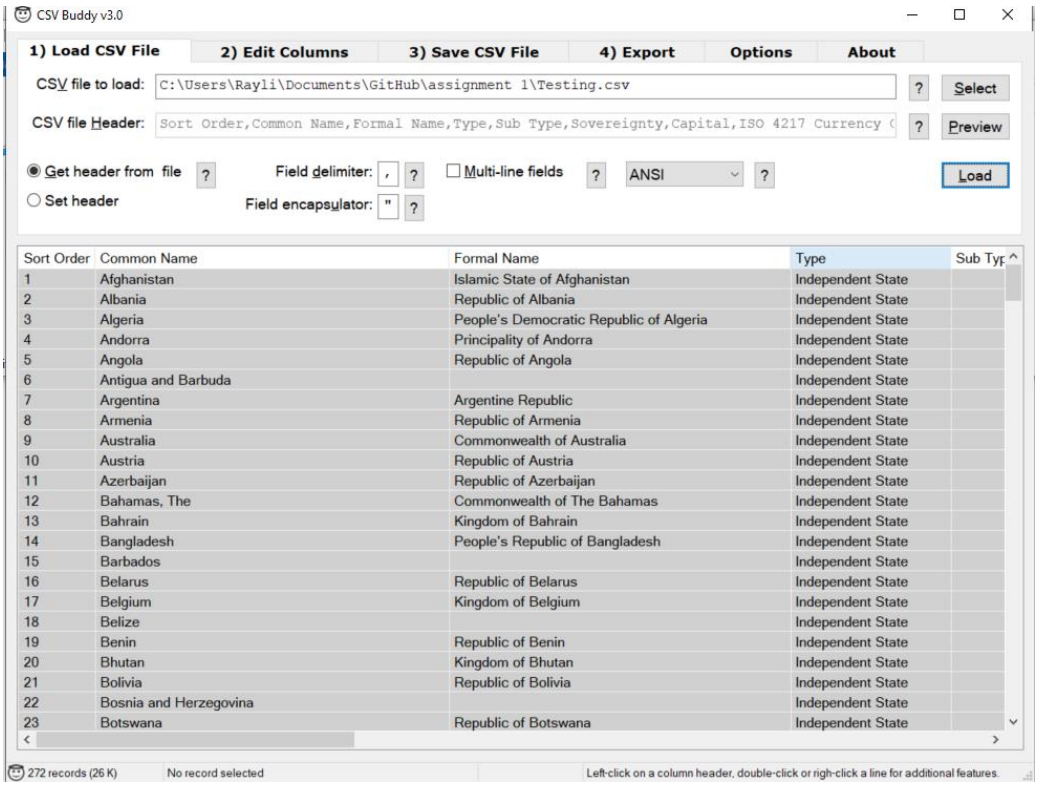
Last, the notification email is sent using the smtplib library, which provides functionality for sending emails via the Simple Mail Transfer Protocol (SMTP). The sender, recipient, subject, and message of the email are defined in variables, and the message variable is constructed with subject and body. The email is sent using the server.sendmail method of the smtplib library, and the connection to the SMTP server is closed using the server.quit method.

3.0 Results

The result of code is that data from a source Country code CSV file is copied into a target testing CSV file, and a notification email is sent to a specified email indicating that the process has completed. The target CSV file will copy the same data as the source CSV file and it will be in a separate file in the same directory. The notification email will contain information about the success copy of the data copying process.



Success email



Copied file

4.0 More works

This code can be extend to do more task, such as additional error handling can be added to the code. It can handle cases where the source or target files are not found, or where there are issues with the data in the source file. For example, the code could be extended to catch specific exceptions and provide more meaningful error messages. It can can be extended to perform more complex data processing on the data from the source file before writing it to the target file. For example, the code could be modified to sort the data, remove duplicates, or perform calculations. This code can solve more extend problem with different requirements and constraints.

REFERENCES

1. *CSV Buddy – Windows freeware download and support*. (n.d.). https://csvbuddy.quickaccesspopup.com/
2. How to explain Robotic Process Automation (RPA) in plain English. (n.d.). *The Enterprisers Project*. https://enterprisersproject.com/article/2019/5/rpa-robotic-process-automation-how-explain