

Email as a Connector

Connector: Email as a Connector

Problem Statement: Reading Email data such as subject, sender email, date, time, email body and attachments.

Official Modules used:

- 1) **Imaplib module in Python:** IMAP is an email retrieval protocol which does not download the emails. It just reads them and displays them. Python's client side library called imaplib is used for accessing emails over imap protocol.
IMAP stands for Internet Mail Access Protocol.
- 2) **Email module in Python:** The email package is an email message management library. The package's main component is an object model that represents email messages.
- 3) **Pandas module in Python:** Pandas is a fast, powerful and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

Process flow/ System flow: Our template consists of 3 files,

- 1) Base.py
- 2) Main.py
- 3) Source_emailconnector.py

Base.py file:

In base.py file, we have our EmailConnector class, which takes username, app password, and IMAP server (ex- Gmail, outlook) as arguments and stores them, also we have "conn" variable initially as None.

Then we have “create_conn” function, which initializes the connection between IMAP server and our user email and then stores our IMAP into “conn” variable.

Source_emailconnector.py:

In this file, we have our “SourceEmailConnector” class which initializes create_conn function with username and app password, then calls “pull_data” function which returns our output as a Data Frame and then calls “__exit__” function to log out of the IMAP server.

- 1) The “Pull_data” function takes ‘N’ as number of emails to be retrieved, and then iterates N times through our email folder (INBOX/SPAM etc), we can also select our email folder using imap.select() function.
- 2) Then while iterating, we parse the bytes email into message object and then decode that message object (here subject, from, date, to) to human-readable Unicode.
- 3) We store all this data into an list each time, and then add each list to final dictionary “dic”.
- 4) Then, we write if else cases for plain text, HTML text(checking for tables), attachments. These 3 cases will be handled in the if-else conditions.
- 5) If we see a plain text then we append that body text into list
- 6) Else if we see a HTML text, we check for tables using pandas “read_html” function which uses “Beautiful Soup” and “Urllib” libraries to extract table data, we then convert all that data table into a data frame and then extract it to the “tables.csv” file.
- 7) Else, if we see any attachments in the email body, we create a folder in that directory and download those attachments and store them in that folder.
- 8) After iterating all the emails, we then convert our dictionary into pandas data frame and then return the final data frame which consists of all the N number of emails subjects, from, date, to and body.

Output:

- 1) "Pull_data" function: Returns a data frame that consists of email subject, date, from, to, body for 'N' number of emails.
- 2) Along with these, if there any attachments present in the email body, they will be downloaded and stored in that folder itself.
- 3) If there are any tables in the form of HTML in email body, that data will be created into a list of tables and stored in a "tables.csv" file in that folder itself.

Example:

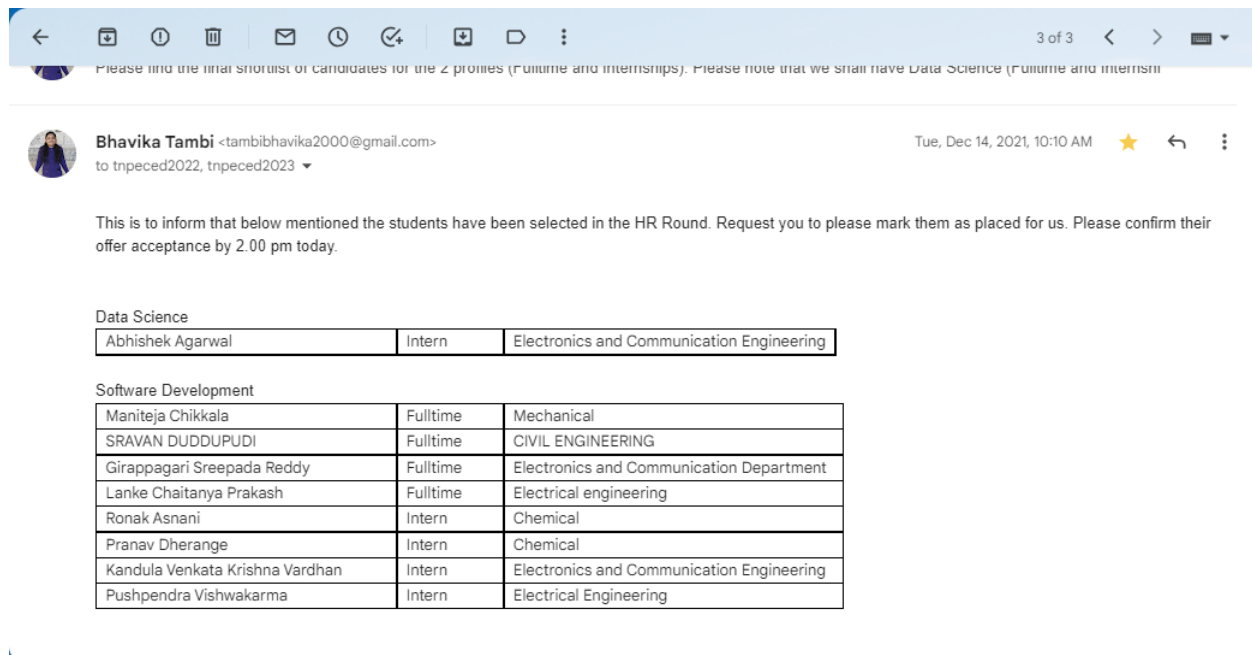
Output of Pull_data function: Returns a data frame that consists of email subject, data, from, to, body for 'N' number of emails.

```
[Running] python -u "c:\Users\mkira\OneDrive\Documents\DPA_Work\Email as a connector\source_emailconnector.py"
| | | | | 0 | | | | | 1
0 Subject This is email subject
1 From Kandula Vardhan <vardhan21kandula@gmail.com>
2 Date Wed, 13 Jul 2022 14:21:06 +0530
3 To kvkvardhan21@gmail.com
4 Email body Decimal Point Analytics is a modern, technolog...

[Done] exited with code=0 in 3.124 seconds
```

This is how the output data frame looks in Visual Studio code.

HTML Tables in email body:



This is the sample input that I have used for reading HTML table inside an email body.

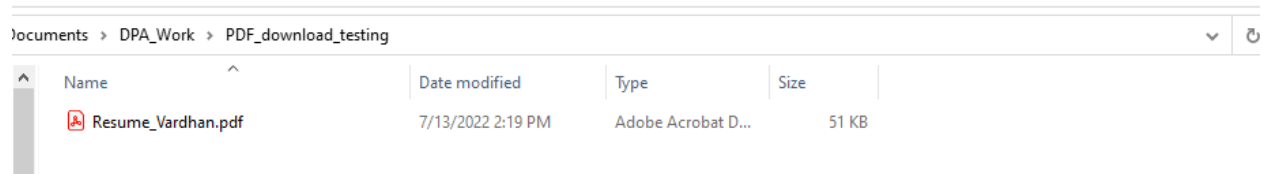
	A	B	C	D	E	F	G	H	I
1		0	1	2 Group					
2		0 Abhishek Agarwal	Intern	Electronics and Communication Engineering	A				
3		0 Maniteja Chikkala	Fulltime	Mechanical	B				
4		1 SRAVAN DUDDUPUDI	Fulltime	CIVIL ENGINEERING	B				
5		2 Girappagari Sreepada Reddy	Fulltime	Electronics and Communication Department	B				
6		3 Lanke Chaitanya Prakash	Fulltime	Electrical engineering	B				
7		4 Ronak Asnani	Intern	Chemical	B				
8		5 Pranav Dherange	Intern	Chemical	B				
9		6 Kandula Venkata Krishna Vardhan	Intern	Electronics and Communication Engineering	B				
10		7 Pushpendra Vishwakarma	Intern	Electrical Engineering	B				
11									
12									
13									
14									

This is how the tables are stored in “tables.csv” file in that same folder itself.


Downloading Attachments:

Name	Date modified	Type	Size
Email as a connector	7/13/2022 2:19 PM	File folder	
PDF_download_testing	7/13/2022 12:21 AM	File folder	
Tables.csv	7/13/2022 2:21 PM	Microsoft Excel C...	1 KB

If we have any attachments, a folder will be created with the email subject as the name of the folder, and the file will be downloaded inside that folder. Here the email subject is “PDF Download testing” and that folder is created for those attachments.



The screenshot shows a Windows File Explorer window with the address bar displaying 'Documents > DPA_Work > PDF_download_testing'. The file list contains one item:

Name	Date modified	Type	Size
 Resume_Vardhan.pdf	7/13/2022 2:19 PM	Adobe Acrobat D...	51 KB

This is the attachment that we have in that email.

References:

About Imaplib module

https://www.tutorialspoint.com/python_network_programming/python_imap.htm#:~:text=Python's%20client%20side%20library%20called,was%20first%20proposed%20in%201986.

About Email module

<https://www.javatpoint.com/email-module-in-python#:~:text=The%20email%20package%20is%20an,model%22%20that%20represents%20email%20messages.>

About Pandas module

<https://pandas.pydata.org/>

Setting up App password in Gmail for reading emails

<https://www.techgeekbuzz.com/blog/how-to-read-emails-in-python/>