**Python Classes Assignment/Questions**

<https://docs.google.com/document/d/1StF2fShh67evZpV9haGYspwYReQSeVjvr2M0ZOiMbDA/edit?tab=t.0>

<https://github.com/Deepanshu-TTN/bootcamp-git/tree/master/week-4/Python%20Classes>

Q1)Write an object oriented code in Python, to download Gmail with attachments .

Create a "Emails.josn" file

In "Emails.josn" file, using Python `pickle` module, dump following information:

- subject

- date

- from email

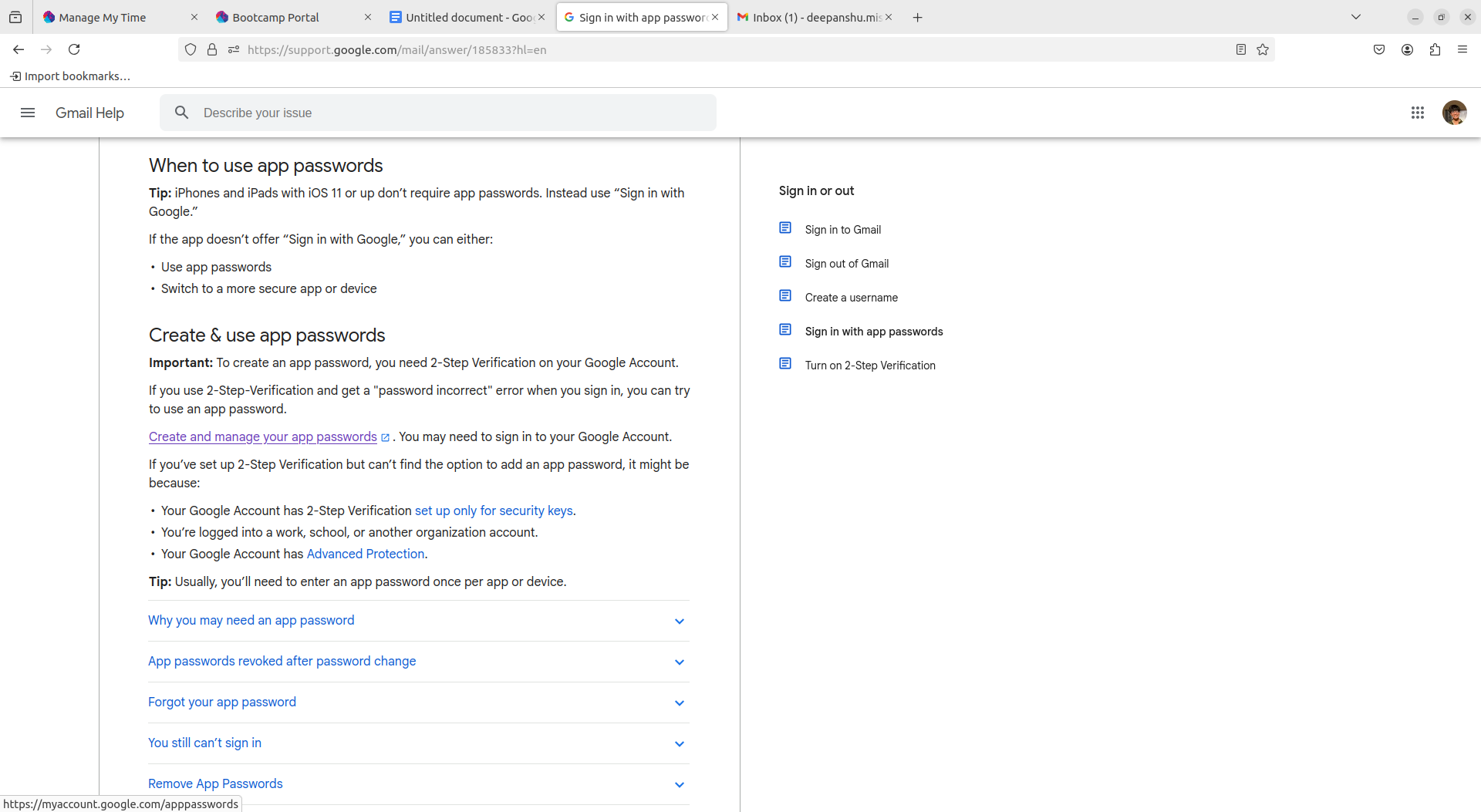
- number of words and line in email body

- number of attachments

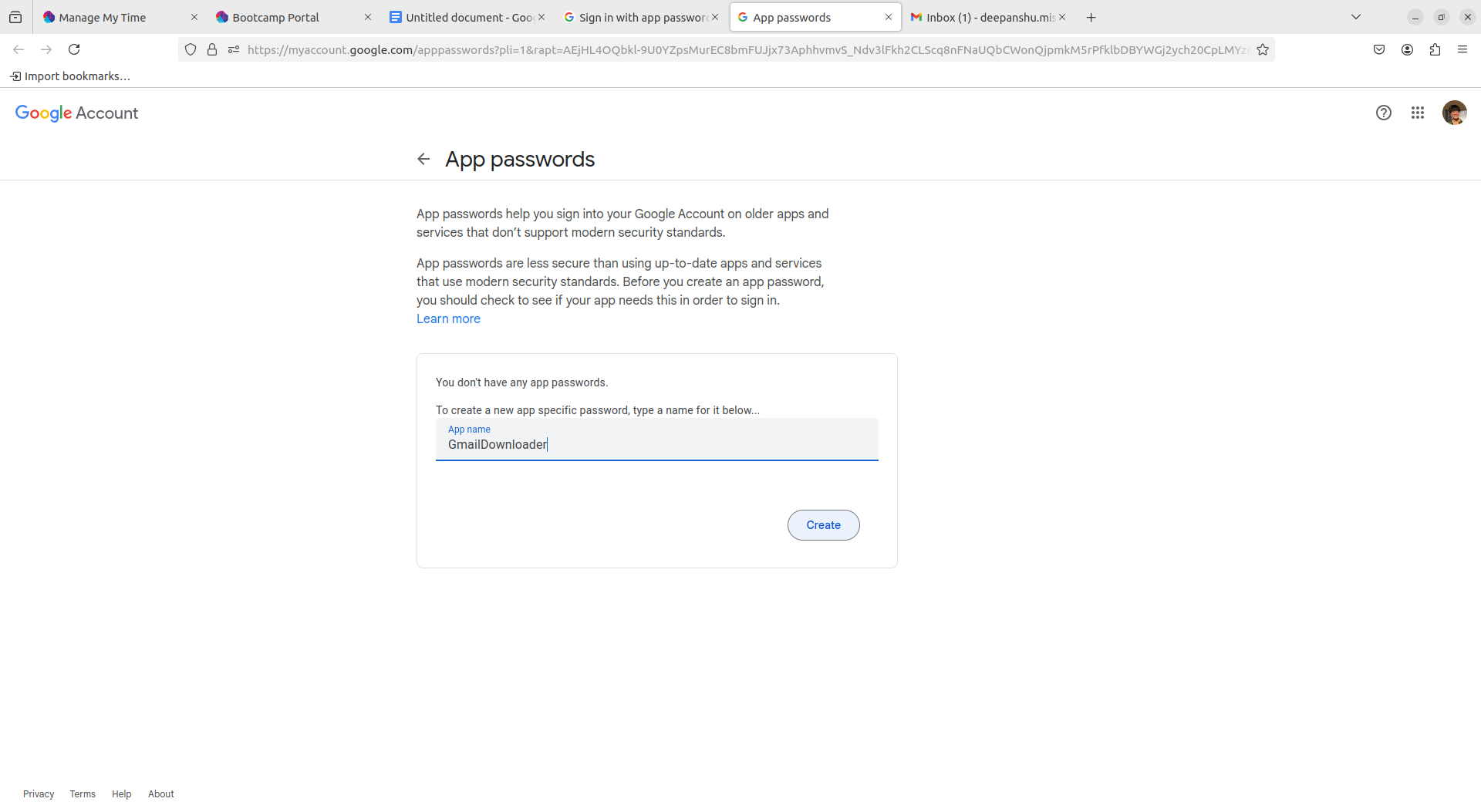
- name of attachments

Each like in "Email.json" should be a valid JSON

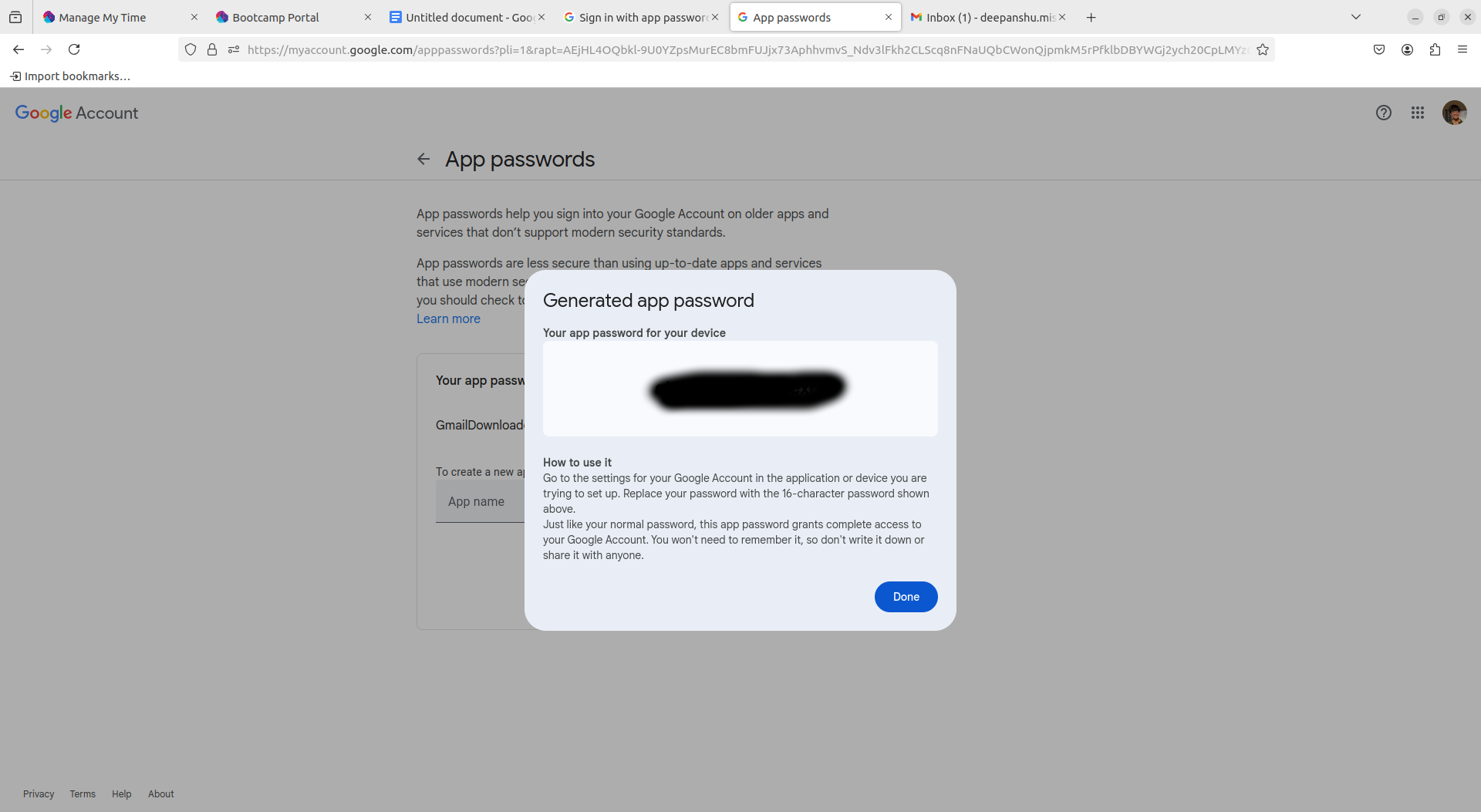
Since I’m planning to use the IMAP package, first need to create an app password to access gmail from my python script, First go to Create and manage your app passwords page:



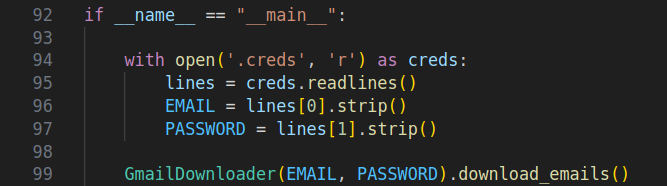
Give the application a suitable name:



Save the generated app password somewhere safe, we will need this later:



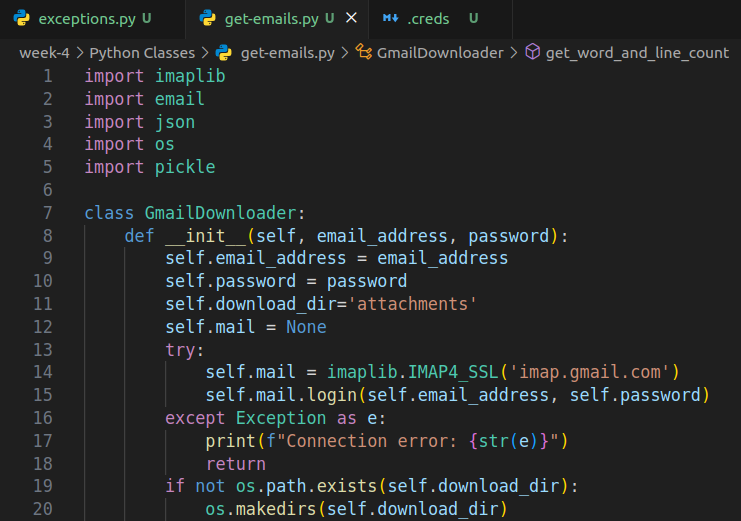
Created a .creds file which stores the email address and this app password which are the two things required by the IMAP package.



GmailDownloader class and **init** function

Here IMAP4\_SSL is a subclass of IMAP4 that specifically creates a secure connection using SSL encryption and ‘imap.gmail.com’ is the hostname for Gmail's IMAP server.

In the init function we are initializing a secure IMAP (Internet Message Access Protocol) connection using the email and the app password generated above. Also to store the fetched attachments, created a new dir in the same directory called attachments



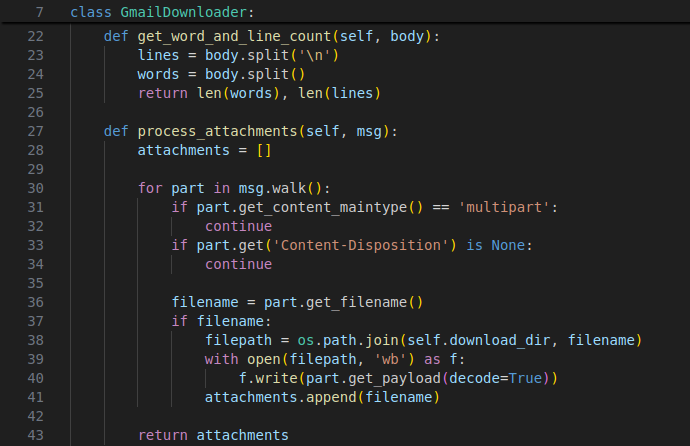
Download\_emails method

This method first selects the inbox to look for mails, then fetches a messages binary string from self.mail and gets the required data from last “max\_emails” mails and dumps it to a emails.json and Emails.pickel file. The for loop iterator ensures that the newest mail is loaded first and theres a break condition when the emails dictionary is long enough and we dont need any more mails we break out of the for loop.

For every mail first we get the email body from msg data. If the message is multipart (or has a parent container which contains plain text, attachments, inline images etc) it searches for the plain text content and inserts it to the body otherwise the whole message is inserted into the body variable. After all the mambo jumbo the data is stored in a dictionary where each email’s binary string is the key and required info is the content and then its dumped to a json and a pickle file



Utility functions get attachments and get count:

In the process attachments function we pass in the raw message from the download mails function, now for every part in this msg (walk function basically is used to discover parts of the mail discussed above), if the part is a multipart container we skip the iteration (since its just a container kindof like a heading) or if there is no Content-Disposition(tells how this part is presented weather its an attachment or inline or none) is none we skip the iteration as well otherwise we get the file and download it in the download\_dir (“attachments” folder in this case) and append the name to the list which is returned after each part is discovered.

In download email assingment,

Provision to download only selective email. For example:

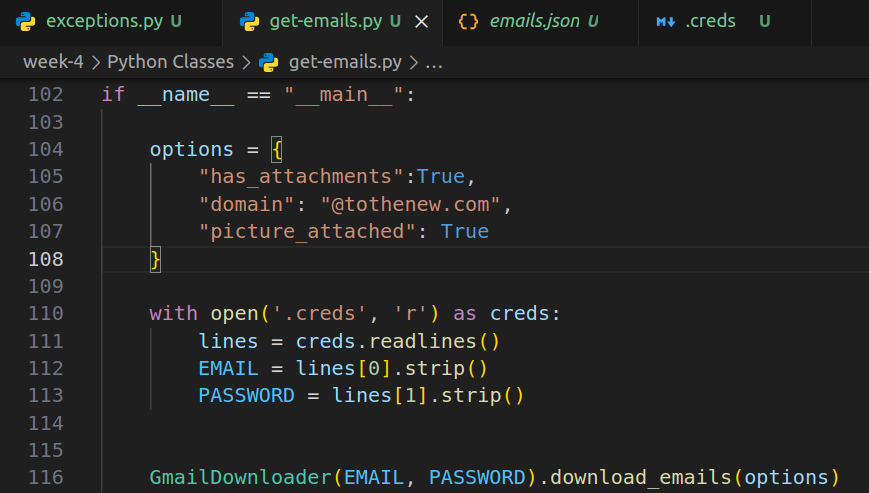
- To download only emails wich have attachements

- To download only email which are related to Job

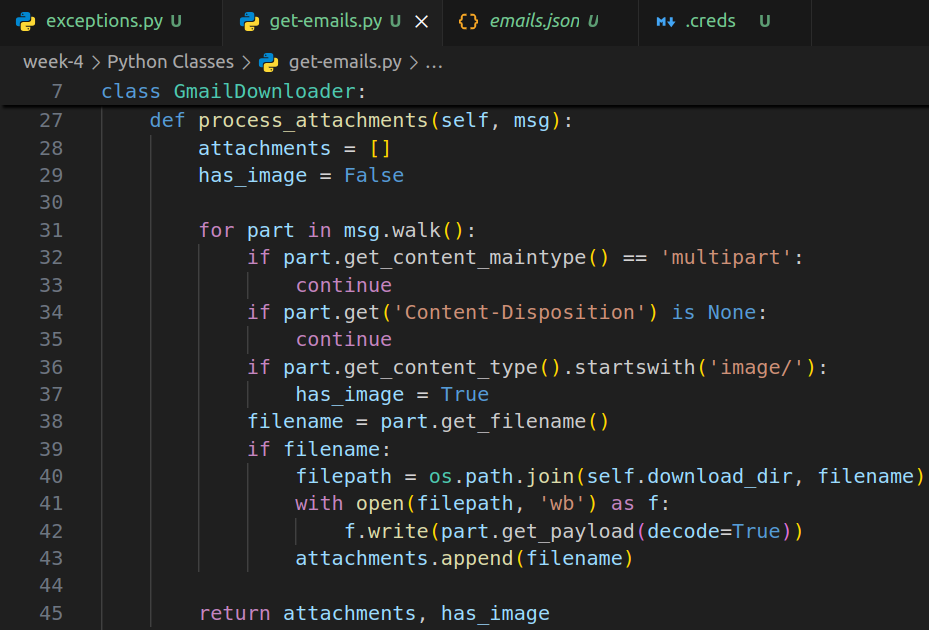
- To download only the email which have picture attached

To achieve this, created an options dictionary in the main part of the code and added an additional parameter for the download email function, also updated the process attachments to return an additional has\_image boolean variable. According to this made changes to the downloader function

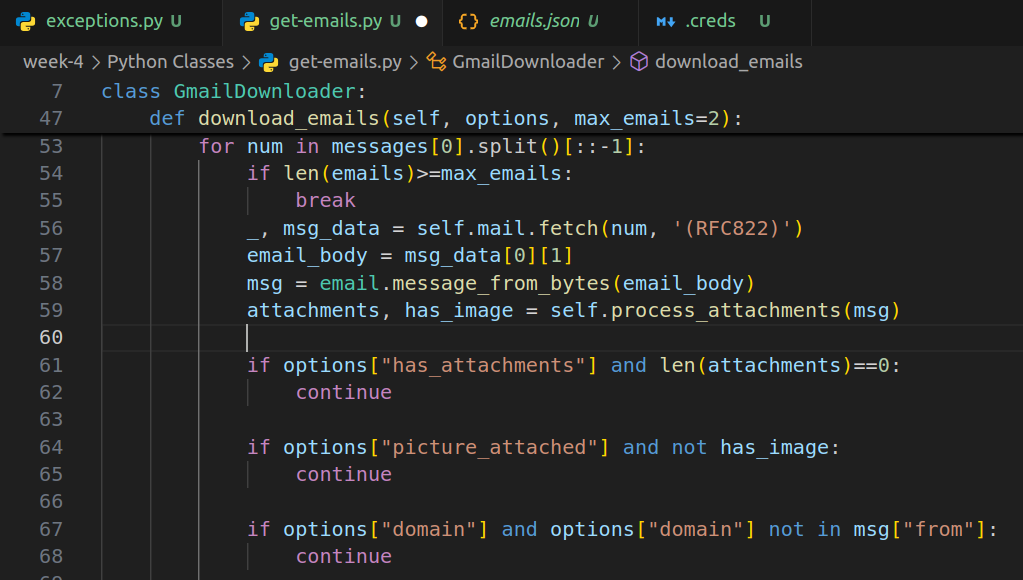
Main part of the code has an options dictionary:



Process\_attachments now returns has\_image flag:



3 additional checks in dictionary formation:



Example output with these options, and max emails=2

