PLACEMENT MESSENGER

A Mini Project Report

B Tech – VI Semester

BATCH1

M.S.K Jahnavi	19331A1236
G.Mythili	19331A1218
M.S.L Bhargavi	19331A1251
P.Tanooj	19331A1240

At

DEPARTMENT OF INFORMATION TECHNOLOGY

MVGR COLLEGE OF ENGINEERING (A)
VIZIANAGARAM.

June 2022

Project Coordinator	Academic Coordinator	HOD - IT
Dr B ANJANADEVI	Dr T PAVAN KUMAR	Dr V NAGESH

ABSTRACT

At present, sometimes students in education institutions are facing problems like delay of receiving information regarding placements and missing the opportunities.To overcome this problem we are going to build a website which can access the administrator g-mail account. This website can read the mails based on dates and can transfer the specific mails to students related to placements.

INTRODUCTION

The main objective of Placement Messenger is to pass the placement information. It manages all the information about placements. The project is totally build at the administrator end and thus only the administrator can access it. The purpose of this project is to build a website to reduce the manual work for passing the information.

PROBLEM STATEMENT

To design a Placement Messenger website which is helpful for passing the information in the educational institution. The implementation of this website is mainly focused on overcoming the delay of information passing present in current system.

SOFTWARE REQUIREMENTS

TKINTER FOR GUI

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. It provides a powerful object-oriented interface to the Tk GUI toolkit. It also provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.

PYTHON

Python is free, open-source, and widely available. Python allows developers to create websites. It requires less coding and makes debugging easier.

EXISTING SYSTEM

In current system, first the information about placements is sent from either Principal Sir or Dean Sir then it is passed HOD Sir of the corresponding department or placement incharge of the department manually. While passing messages manually, sometimes it may leads to delay and the students may miss the opportunity.

PROPOSED SYSTEM

The main task of this system is to reduce the manual work. In this Placement Messenger, the administrator must login to the website. So that the system can access the mails. In the proposed system, the information passing done automatically by accessing the message which is in the administrator mail.

In this system we used imaplib and smtplib modules:

imaplib module:

imaplib is a Python module or library that provide us with client classes so that we can set up communication with IMAP version 4 servers, and through this IMAP communication, we can retrieve data from our emails. imaplib library provides us with three client classes that are used while communicating with the servers using the IMAP protocol in Python.

- IMAP_4
- IMAP4_Stream
- IMAP_SSL

These classes of imaplib module are used to set up a communication with the server while we are using the IMAP protocol to access our emails' data through a Python program.

IMAP protocol has several different commands which are used to perform several different actions through it. Using these commands of IMAP protocol, we can perform multiple actions on our email box, and thus these commands help us to retrieve information from our emails.

LOGIN

This command is used to open the connection with the email server by logging into the server through the credentials we will provide.

SELECT

We use this command to select the mailbox folder which mail we want to access, and thus it accesses all the emails present in the mailbox. We can even make changes in the mailbox after selecting the mailbox through this SELECT command.

LOGOUT

When we are not using the IMAP protocol or when we have done our work with the emails, then we want to close the connection with the email server, and we can do this using the LOGOUT command. This command informs the email server that the user is done with the session, and now the session should be closed. The server will first send the BYE response through the protocol, followed by the OK response from the client-side and then the connection with the server will be closed.

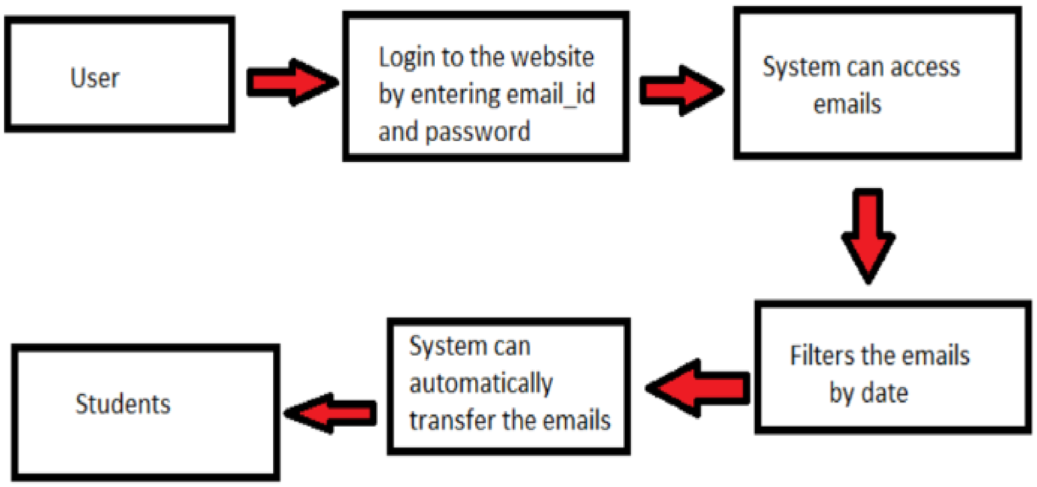
smaplib module:

Simple Mail Transfer Protocol (SMTP) is used as a protocol to handle the email transfer using Python. It is used to route emails between email servers. It is an application layer protocol which allows to users to send mail to another.

It accepts the following parameters.

- **host:** It is the hostname of the machine which is running your SMTP server. Here, we can specify the IP address of the server like (https://www.javatpoint.com) or localhost. It is an optional parameter.
- **port:** It is the port number on which the host machine is listening to the SMTP connections. It is 25 by default.
- **local_hostname:** If the SMTP server is running on your local machine, we can mention the hostname of the local machine.

PROCESS FLOW


User
Login to the website by entering email_id and password
System can access emails
Filters the emails by date
System can automatically transfer the emails
Students

First the user has to login to the website by entering their email id and password, then the system will access the emails and also filters the emails by date. After filtering the system can automatically the emails to students.

PSEUDO CODE

def getAllEmails(username, password, folderName)

- This function is used to send all the mails without any filters.
- Inputs of this function are username, password of the admin to login and foldername to save the transferred mails in the folder
- This function executes and transfer all mails in the admin e-mail to student emails

def getMailsUsingDate(username, password, year, month, date, folderName)

- This function is used to filter the mails based on the date and send them.
- Inputs of the function are username, password of the admin to login and year,month,date are used to get the date
- This function executes and transfer mails by filtering through dates and send them to student emails

def getMailsUsingSender(username, password, fromEmail, folderName)

- This funtion is used to filter the mails based on particular sender's mail id
- Inputs of the function are username, password of the admin to login and fromEmail of the particular email id
- This function executes and transfers all the mails according to sender email id

def getMailsUsingDateAndSender(username, password, year, month, date, fromEmail, folderName)

- This function is used to filter the mails based on dates and particular sender's mail id

- Inputs of the function are username , password of the admin to login, year, month & date for filtering based on date ,sender’s email id as fromEmail and folder name to save the details of the mails which are transferred.

IMPLIMENTATION:

```
from tkinter import *

from functools import partial

#from pro import *

import imaplib

import smtplib

import email

import os

from datetime import datetime

import mimetypes

import smtplib

from getpass import getpass


#window

tkWindow = Tk()

tkWindow.geometry('400x400')

tkWindow.title('Login Form ')

tkWindow['background']='cyan'

#Title

titleLabel = Label(tkWindow, text=" Placement Messenger ",bg='lawn green').grid(row=0, column=1 ,columnspan=2)
```

```
def validateLogin():
   chWindow = Toplevel(tkWindow)
   chWindow.geometry('400x400')
   chWindow.title('Choice Form')
   chWindow['background']='cyan'
   chTitle=Label(chWindow,text="Choose your choice",bg="peach puff").grid(row=0,column=1)
   chLabel3 = Label(chWindow,text="Filter through Sender : ").grid(row=2,column=0)
   chButton3 = Button(chWindow,text="Filter through Sender",command=SendEmails).grid(row=2,column=1)
   chLabel4 = Label(chWindow,text="Filter throuht both Sender and Date : ").grid(row=4,column=0)
   chButton4 = Button(chWindow,text="Filter throuht both Sender and Date",command=SDEmails).grid(row=4,column=1)
def AllEmails():
   global Folder
   allWindow = Tk()
   allWindow.geometry('400x400')
   allWindow.title('AllEmails Form')
   alltitle = Label(allWindow, text=" Placement Messenger ").grid(row=0, column=1 ,columnspan=2)
   allFolder = Label(allWindow, text="  Folder  :").grid(row=1, column=0)
   #Folder = StringVar()

   Folder = Entry(allWindow)
   Folder.grid(row=1, column=1)
```

```
    allLabel = Label(allWindow, text="   All mails are been sent ").grid(row=2, column=1)

    #allButton = Button(allWindow, text="submit", command=lambda:button(Folder.get())).grid(row=5, column=1,columnspan=2)

    allButton = Button(allWindow, text="submit", command=sddt).grid(row=5, column=1,columnspan=2)


def button(e):

    print(e)

def DateEmails():

    global Folder

    global Year

    global Month

    global Date

    dateWindow = Tk()

    dateWindow.geometry('400x400')

    dateWindow.title('Emails Filtering Through Date Form')

    datetitle = Label(dateWindow, text=" Filtering through Date ").grid(row=0, column=1 ,columnspan=2)

    dateFolderLabel = Label(dateWindow, text="  Folder  :").grid(row=1, column=0)

    #Folder = StringVar()

    Folder = Entry(dateWindow)

    Folder.grid(row=1, column=1)


    dateYearLabel= Label(dateWindow, text="   Year   :").grid(row=2, column=0)
    #Year = StringVar()
    Year = Entry(dateWindow)
    Year.grid(row=2, column=1)

    dateMonthLabel = Label(dateWindow, text="   Month(numbers only)
:").grid(row=3, column=0)
    #Month= StringVar()
    Month = Entry(dateWindow)
    Month.grid(row=3, column=1)

    dateDateLabel = Label(dateWindow, text="   Date   :").grid(row=4,
column=0)
    #Date= StringVar()
    Date= Entry(dateWindow)
    Date.grid(row=4, column=1)



    dateloginButton = Button(dateWindow, text="submit").grid(row=5,
column=1,columnspan=2)
def SendEmails():
    global Folder
    global Sender
    sendWindow = Tk()
    sendWindow.geometry('400x400')
    sendWindow.title('Emails Filtering Through Sender Form')
    sendWindow['background']='cyan'
    sendtitle = Label(sendWindow, text=" Filtering through Sender Email ",bg="bisque2").grid(row=0, column=1 ,columnspan=2)

    sendFolderLabel = Label(sendWindow, text="  Folder  :").grid(row=1, column=0)

    #Folder = StringVar()

    Folder= Entry(sendWindow)

    Folder.grid(row=1, column=1)


    SenderLabel = Label(sendWindow, text="    Sender(emailId) :").grid(row=2, column=0)

    #Sender = StringVar()

    Sender = Entry(sendWindow)

    Sender.grid(row=2, column=1)

    sendloginButton = Button(sendWindow, text="submit",bg="yellow",command=dt).grid(row=5, column=1,columnspan=2)


def SDEmails():

    global Folder

    global Year

    global Month

    global Date

    global Sender

    sdWindow = Tk()

    sdWindow.geometry('400x400')

    sdWindow.title('Emails Filtering Through Date and Sender Form')

    sdWindow['background']='cyan'
```

```
titleLabel = Label(sdWindow, text=" Filtering through Sender & Date ",bg='LemonChiffon2').grid(row=0, column=1 ,columnspan=2)

#username label and text entry box

sdFolderLabel = Label(sdWindow, text="  Folder  :").grid(row=2, column=0)

#global Folder

#Folder = StringVar()

#global sdFolderEntry

Folder = Entry(sdWindow)

Folder.grid(row=2, column=1)

#print(Folder.get())



sdYearLabel = Label(sdWindow, text="  Year  :").grid(row=4, column=0)

#global Year

#Year = StringVar()

Year= Entry(sdWindow)

Year.grid(row=4, column=1)


sdMonthLabel = Label(sdWindow, text="  Month(numbers only) :").grid(row=6, column=0)

#global Month

#Month= StringVar()

Month= Entry(sdWindow)

Month.grid(row=6, column=1)


sdDateLabel = Label(sdWindow, text="  Date  :").grid(row=8, column=0)
```

```
    #global Date
    #Date= StringVar()
    Date= Entry(sdWindow)
    Date.grid(row=8, column=1)

    sdSenderLabel = Label(sdWindow, text=" Sender(emailId)
:").grid(row=10, column=0)
    #global Sender
    #Sender = StringVar()
    Sender = Entry(sdWindow)
    Sender.grid(row=10, column=1)
    sdloginButton = Button(sdWindow, text="click on
me!",bg="yellow",command=sddt).grid(row=12,
column=1,columnspan=2)

def sddt():
    global Folder
    Username=username.get()
    Password=password.get()

    folder=Folder.get()
    year=Year.get()
    month=Month.get()
    date=Date.get()
    sender=Sender.get()

    #print(username,password)
    #print(Folder,Year,Month,Date,Sender)
    #print(sdFolderEntry.get())
    year=int(year)
    month=int(month)
    date=int(date)
    #print(type(folder))
    ##print(type(date))
    #print(type(sender))
    #print(Username,Password,year,month,date,sender,folder)

getMailsUsingDateAndSender(Username,Password,year,month,date,send
er,folder)
    print("Mail has been send to students based on date given",date,"-
",month,"-",year,"and sender is",sender)
def getMailsUsingDateAndSender(Username, Password, year, month,
date, fromEmail, folderName):
    #print(Username,Password,year,month,date,fromEmail,folderName)
    mail = imaplib.IMAP4_SSL("imap.gmail.com")
    #print("into the  function")
    mail.login(Username, Password)
    print("Login success..........")


    mail.select("inbox")
    #year=int(year)
    #date=int(date)
    #month=int(month)
    # querying through search method to filter emails based on date we provided.
    x1 = datetime(year, month, date)
    startDate = x1.strftime("%d-%b-%Y")
    result, data = mail.search(None, '(SENTSINCE {0})'.format(startDate))
    inbox_item_list_date = data[0].split()


    # querying through search method to filter emails based on sender mail we provided.
    result, data = mail.search(None, 'FROM', '"{}"'.format(fromEmail))
    inbox_item_list_sender = data[0].split()


    #We take intersection of these sets so that we have UIDs of only those which satify both criteria.
    inbox_item_list = list(set(inbox_item_list_date) & set(inbox_item_list_sender))


    counter = 0
    for item in inbox_item_list:
        counter+=1
        result2, email_data = mail.fetch(item,'(RFC822)')
        raw_email = email_data[0][1].decode("utf-8")


        email_message = email.message_from_string(raw_email)


        to_ = email_message['To']
        from_ = email_message['From']
```

```
subject_ = email_message['Subject']
date_ = email_message['date']
sub1 = subject_
d = date_

to_ = "to: " + to_ + str("\n")
from_ = "from: " + from_ + str("\n")
date_ = "date: " + date_ + str("\n")
subject__ = "subject: " + subject_ + str("\n")

lenOfSubject = len(subject_)
if (lenOfSubject > 30):
    subject_ = "exceed"+str(counter)

for part in email_message.walk():
    if part.get_content_maintype == 'multipart':
        continue
    content_type = part.get_content_type()
    content_disposition = str(part.get("Content-Disposition"))

    filename = part.get_filename()

    ext = mimetypes.guess_extension(part.get_content_type())
    if ext == '.pdf' or ext == '.jpe' or ext == '.png' or ext == '.docx':

        if filename:
```

```
                    save_path = os.path.join(os.getcwd(), folderName, subject_)

                    if not os.path.exists(save_path):
                        os.makedirs(save_path)
                    with open(os.path.join(save_path, filename), 'wb') as fp:
                        fp.write(part.get_payload(decode=True))
                        fp.close()


            try:
                body = part.get_payload(decode=True).decode()
            except:
                pass

            if content_type == "text/plain" and "attachment" not in
content_disposition:
                save_path = os.path.join(os.getcwd(), folderName, subject_)

                if not os.path.exists(save_path):
                    os.makedirs(save_path)


                filename = "textfile.txt"
                with open(os.path.join(save_path, filename), 'w+',
encoding='utf-8') as fp:
```

```
                fp.writelines(to_)
                fp.writelines(from_)
                fp.writelines(date_)
                fp.writelines(subject__)
                fp.writelines(body)
                fp.close()
            server = smtplib.SMTP_SSL("smtp.gmail.com",465)
            server.login(Username,Password)
            msg=subject__ +"\n"+body

to=['jahnavimulaga123@gmail.com','jaswanthimulaga121@gmail.com','v
ennelaranikuna@gmail.com']
            server.sendmail(Username,to,msg)
            server.quit()
    mail.close()
    mail.logout()
def dt():
    Username=username.get()
    Password=password.get()
    folder=Folder.get()
    sender=Sender.get()
    getMailsUsingSender(Username, Password, sender, folder)
    print("All filtered mails based sender",sender,"has been sent to
students")
def getMailsUsingSender(username, password, fromEmail, folderName):
    mail = imaplib.IMAP4_SSL("imap.gmail.com")
    mail.login(username, password)
    print("Login success.........")


    mail.select("inbox")
    # querying through search method to filter emails based on sender mail
we provided.
    result, data = mail.search(None, 'FROM', '"{}"'.format(fromEmail))
    inbox_item_list = data[0].split()
    counter = 0
    for item in inbox_item_list:
        counter+=1
        result2, email_data = mail.fetch(item,'(RFC822)')
        raw_email = email_data[0][1].decode("utf-8")


        email_message = email.message_from_string(raw_email)


        to_ = email_message['To']
        from_ = email_message['From']
        subject_ = email_message['Subject']
        date_ = email_message['date']


        to_ = "to: " + to_ + str("\n")
        from_ = "from: " + from_ + str("\n")
        date_ = "date: " + date_ + str("\n")
        subject__ = "subject: " + subject_ + str("\n")


        lenOfSubject = len(subject_)
```

```
if (lenOfSubject > 30):
    subject_ = "exceed"+str(counter)
    print(subject_)

for part in email_message.walk():
    if part.get_content_maintype == 'multipart':
        continue
    content_type = part.get_content_type()
    content_disposition = str(part.get("Content-Disposition"))

    filename = part.get_filename()

    ext = mimetypes.guess_extension(part.get_content_type())
    if ext == '.pdf' or ext == '.jpe' or ext == '.png' or ext == '.docx':

        if filename:

            save_path = os.path.join(os.getcwd(), folderName, subject_)

            if not os.path.exists(save_path):
                os.makedirs(save_path)
            with open(os.path.join(save_path, filename), 'wb') as fp:
                fp.write(part.get_payload(decode=True))
                fp.close()
        try:
            body = part.get_payload(decode=True).decode()

        except:
            pass

        if content_type == "text/plain" and "attachment" not in
content_disposition:
            save_path = os.path.join(os.getcwd(), folderName, subject_)

            if not os.path.exists(save_path):
                os.makedirs(save_path)

            filename = "textfile.txt"
            with open(os.path.join(save_path, filename), 'w+',
encoding='utf-8') as fp:
                fp.writelines(to_)
                fp.writelines(from_)
                fp.writelines(date_)
                fp.writelines(subject__)
                fp.writelines(body)
                fp.close()
            server = smtplib.SMTP_SSL("smtp.gmail.com",465)
            server.login(username,password)
            msg=subject__ +"\n"+body

to=['jahnavimulaga123@gmail.com','jaswanthimulaga121@gmail.com']
```

```
            server.sendmail(username,to,msg)

            server.quit()


    mail.close()

    mail.logout()

def getAllEmails(username, password, folderName):

    # used to make an connection over imap4 server over an SSL encrypted
socket

    # in our case that server is gmail

    # If port is omitted, the standard IMAP4-over-SSL port (993) is used

    mail = imaplib.IMAP4_SSL("imap.gmail.com")

    # login is used to identify client

    mail.login(username, password)

    print("Login success..........")


    # we can select any directory using mail.list(), in our case we have
selected inbox.

    mail.select("inbox")


    # mails are identified by UID number

    result, data = mail.uid('search',None,'ALL')


    #This is a list containing UID number for each mail present in Inbox
mail.

    inbox_item_list = data[0].split()


    counter = 0
```

```
 # iterating over UIDs
 for item in inbox_item_list:
     counter+=1
     #result2 contains confirmation in the form of "OK" and email_data
contains information regarding the mail.
     result2, email_data = mail.uid('fetch',item,'(RFC822)')


     raw_email = email_data[0][1].decode("utf-8")


     #Return a message object structure from a string.
     email_message = email.message_from_string(raw_email)


     #getting information about the mail like to, from,subject, date.
     to_ = email_message['To']
     from_ = email_message['From']
     subject_ = email_message['Subject']
     date_ = email_message['date']


     # setting the format to save in text file.
     to_ = "to: " + to_ + str("\n")
     from_ = "from: " + from_ + str("\n")
     date_ = "date: " + date_ + str("\n")
     subject__ = "subject: " + subject_ + str("\n")
```

```
    # if path length exceeds a certain limit, then changing the name of
mail folder.
    lenOfSubject = len(subject_)
    if (lenOfSubject > 30):
      #Setting subject equals to exceed + counter if len of subject is more
than 30.
      subject_ = "exceed"+str(counter)


    # accessing the subparts of email_message
    for part in email_message.walk():
      if part.get_content_maintype == 'multipart':
        continue
      content_type = part.get_content_type()
      content_disposition = str(part.get("Content-Disposition"))


      filename = part.get_filename()
      # using mimetype to know the extension of attachment
      # comment below 2 lines to allow all types of format to download
in all functions.
      ext = mimetypes.guess_extension(part.get_content_type())
      # allowing pdf, jpg, png and doc format only
      if ext == '.pdf' or ext == '.jpe' or ext == '.png' or ext == '.docx':
        if filename:
          save_path = os.path.join(os.getcwd(), folderName, subject_)
          if not os.path.exists(save_path):
            os.makedirs(save_path)
          with open(os.path.join(save_path, filename), 'wb') as fp:
            fp.write(part.get_payload(decode=True))
            fp.close()

    # getting the body part of the mail.
      try:
        body = part.get_payload(decode=True).decode()
      except:
        pass

    # saving the required information in a file named as "textfile.txt".
      if content_type == "text/plain" and "attachment" not in
content_disposition:
        save_path = os.path.join(os.getcwd(), folderName, subject_)

        if not os.path.exists(save_path):
          os.makedirs(save_path)

        filename = "textfile.txt"
        with open(os.path.join(save_path, filename), 'w+',
encoding='utf-8') as fp:
          fp.writelines(to_)
          fp.writelines(from_)
          fp.writelines(date_)
          fp.writelines(subject__)
          fp.writelines(body)    #Add here if any other information you
want to add in text file.
          fp.close()
```

```
            server = smtplib.SMTP_SSL("smtp.gmail.com",465)
            server.login(username,password)
            msg=subject__ +"\n"+body
            to=['miniproject296@gmail.com']
            server.sendmail(username,to,msg)
            server.quit()
    mail.close()
    mail.logout()
def getMailsUsingDate(username, password, year, month, date, folderName):
    mail = imaplib.IMAP4_SSL("imap.gmail.com")
    mail.login(username, password)
    print("Login success..........")

    mail.select("inbox")

    # seeting the year, month, date in strftime format.
    x1 = datetime(year, month, date)
    startDate = x1.strftime("%d-%b-%Y")
    # querying through search method to filter emails based on date we provided.
    result, data = mail.search(None, '(SENTSINCE {0})'.format(startDate))
    inbox_item_list = data[0].split()

    counter = 0
    for item in inbox_item_list:
```

```
counter+=1
result2, email_data = mail.fetch(item,'(RFC822)')
raw_email = email_data[0][1].decode("utf-8")

email_message = email.message_from_string(raw_email)

to_ = email_message['To']
from_ = email_message['From']
subject_ = email_message['Subject']
date_ = email_message['date']

to_ = "to: " + to_ + str("\n")
from_ = "from: " + from_ + str("\n")
date_ = "date: " + date_ + str("\n")
subject__ = "subject: " + subject_ + str("\n")

lenOfSubject = len(subject_)
if (lenOfSubject > 30):
    subject_ = "exceed"+str(counter)

for part in email_message.walk():
    if part.get_content_maintype == 'multipart':
        continue
    content_type = part.get_content_type()
    content_disposition = str(part.get("Content-Disposition"))
```

```
filename = part.get_filename()


ext = mimetypes.guess_extension(part.get_content_type())
if ext == '.pdf' or ext == '.jpe' or ext == '.png' or ext == '.docx':

    if filename:

        save_path = os.path.join(os.getcwd(), folderName, subject_)

        if not os.path.exists(save_path):
            os.makedirs(save_path)
        with open(os.path.join(save_path, filename), 'wb') as fp:
            fp.write(part.get_payload(decode=True))
            fp.close()



try:
    body = part.get_payload(decode=True).decode()

except:
    pass

if content_type == "text/plain" and "attachment" not in
content_disposition:
    save_path = os.path.join(os.getcwd(), folderName, subject_)
```

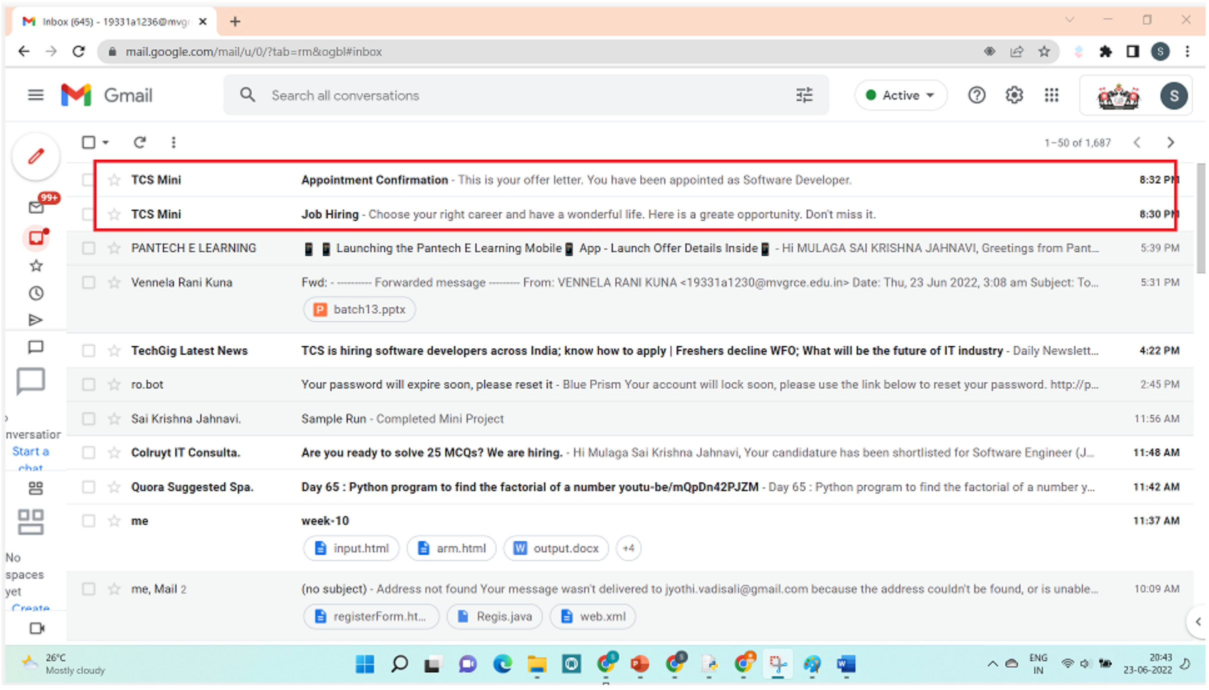
```
            if not os.path.exists(save_path):
                os.makedirs(save_path)


            filename = "textfile.txt"
            with        open(os.path.join(save_path,        filename),        'w+',
encoding='utf-8') as fp:
                fp.writelines(to_)
                fp.writelines(from_)
                fp.writelines(date_)
                fp.writelines(subject__)
                fp.writelines(body)
                fp.close()
            server = smtplib.SMTP_SSL("smtp.gmail.com",465)
            server.login(username,password)
            msg=subject__ +"\n"+body

to=['jahnavimulaga123@gmail.com','jaswanthimulaga121@gmail.com']
            server.sendmail(username,to,msg)
            server.quit()


   mail.close()
   mail.logout()




#username label and text entry box
```

```
usernameLabel = Label(tkWindow, text="   User  Name   :").grid(row=1, column=0)

#username=StringVar()

#username

username = Entry(tkWindow)

username.grid(row=1, column=1)


#password label and password entry box

passwordLabel = Label(tkWindow,text="    Password     :").grid(row=4, column=0)

#password = StringVar()

#global password

password = Entry(tkWindow, show='*')

password.grid(row=4, column=1)

global Folder

global Year

global Month

global Date

global Sender



#validateLogin = partial(validateLogin, username, password)


#login button

loginButton = Button(tkWindow, text="Login",bg="yellow" ,command=validateLogin).grid(row=5, column=1,columnspan=2)
```

```
#page2
tkWindow.mainloop()
```

RESULT AND ANALYSIS

Admin Email Inbox:



Inbox (645) - 19331a1236@mvg

mail.google.com/mail/u/0/?tab=rm&ogbl#inbox

Search all conversations

1–50 of 1,687

TCS Mini — **Appointment Confirmation** - This is your offer letter. You have been appointed as Software Developer. — 8:32 PM

TCS Mini — **Job Hiring** - Choose your right career and have a wonderful life. Here is a greate opportunity. Don't miss it. — 8:30 PM

PANTECH E LEARNING — Launching the Pantech E Learning Mobile App - Launch Offer Details Inside - Hi MULAGA SAI KRISHNA JAHNAVI, Greetings from Pant... — 5:39 PM

Vennela Rani Kuna — Fwd: - --------- Forwarded message -------- From: VENNELA RANI KUNA <19331a1230@mvgrce.edu.in> Date: Thu, 23 Jun 2022, 3:08 am Subject: To... — 5:31 PM — batch13.pptx

TechGig Latest News — **TCS is hiring software developers across India; know how to apply | Freshers decline WFO; What will be the future of IT industry** - Daily Newslett... — 4:22 PM

ro.bot — Your password will expire soon, please reset it - Blue Prism Your account will lock soon, please use the link below to reset your password. http://p... — 2:45 PM

Sai Krishna Jahnavi. — Sample Run - Completed Mini Project — 11:56 AM

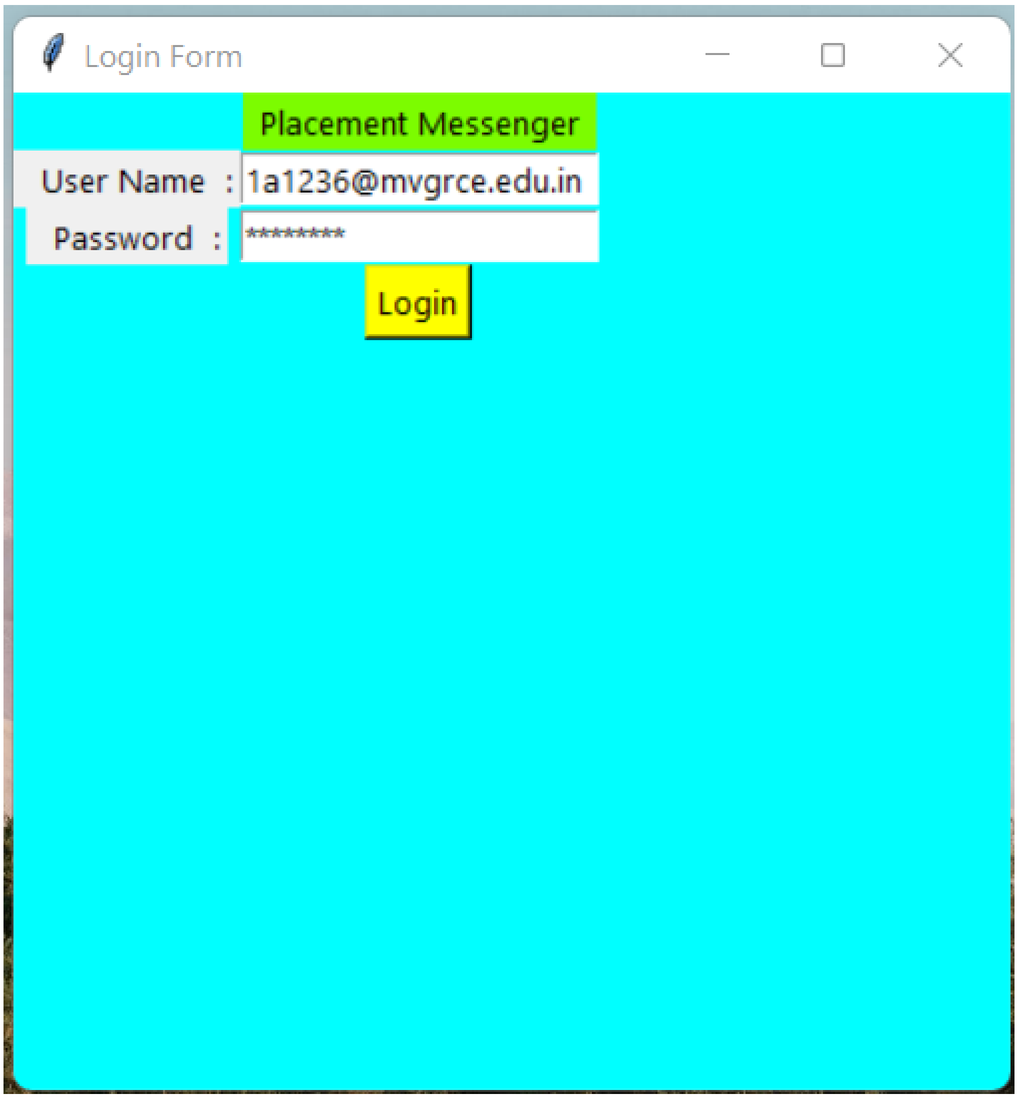
Colruyt IT Consulta. — **Are you ready to solve 25 MCQs? We are hiring.** - Hi Mulaga Sai Krishna Jahnavi, Your candidature has been shortlisted for Software Engineer (J... — 11:48 AM

Quora Suggested Spa. — **Day 65 : Python program to find the factorial of a number youtu-be/mQpDn42PJZM** - Day 65 : Python program to find the factorial of a number y... — 11:42 AM

me — **week-10** — 11:37 AM — input.html, arm.html, output.docx, +4

me, Mail 2 — (no subject) - Address not found Your message wasn't delivered to jyothi.vadisali@gmail.com because the address couldn't be found, or is unable... — 10:09 AM — registerForm.ht..., Regis.java, web.xml

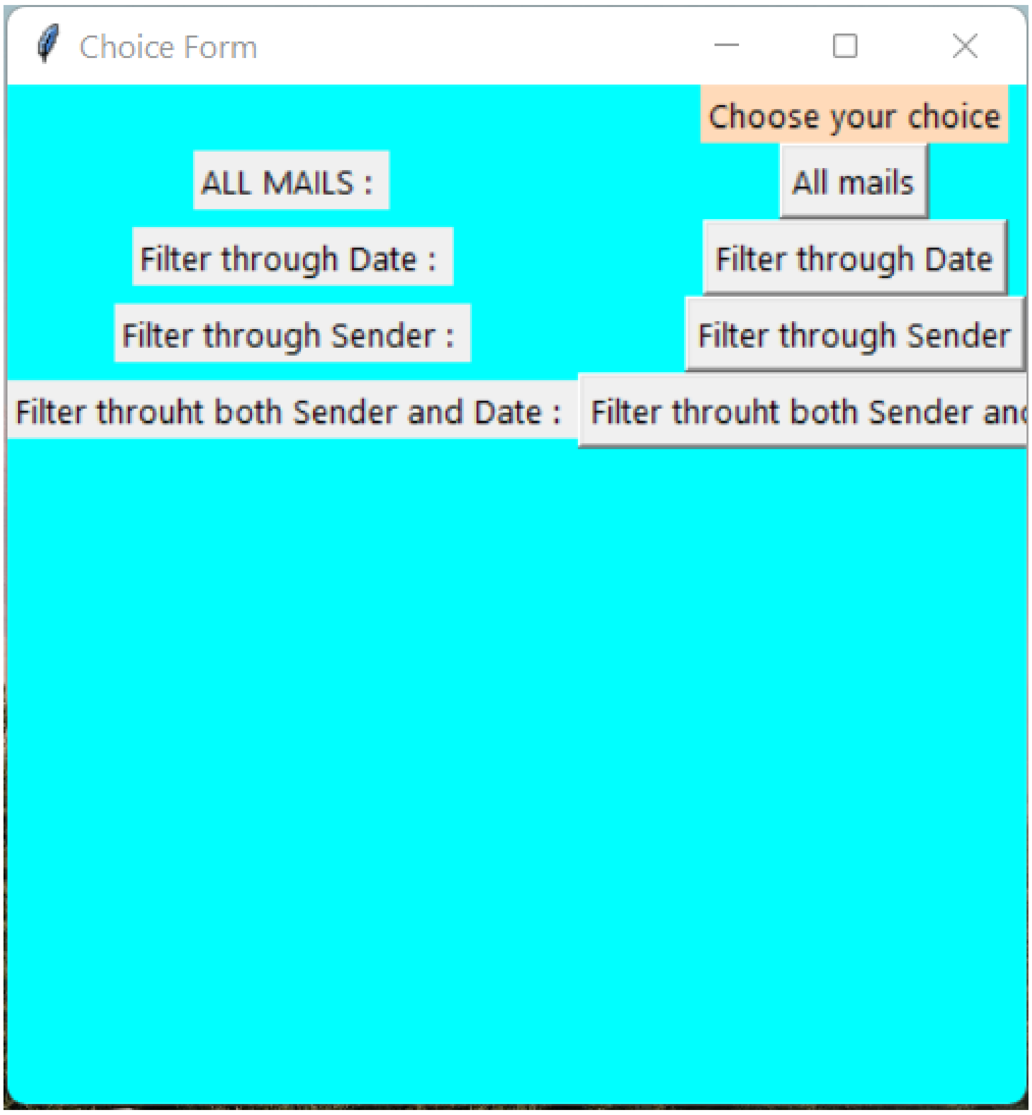
Admin Login:



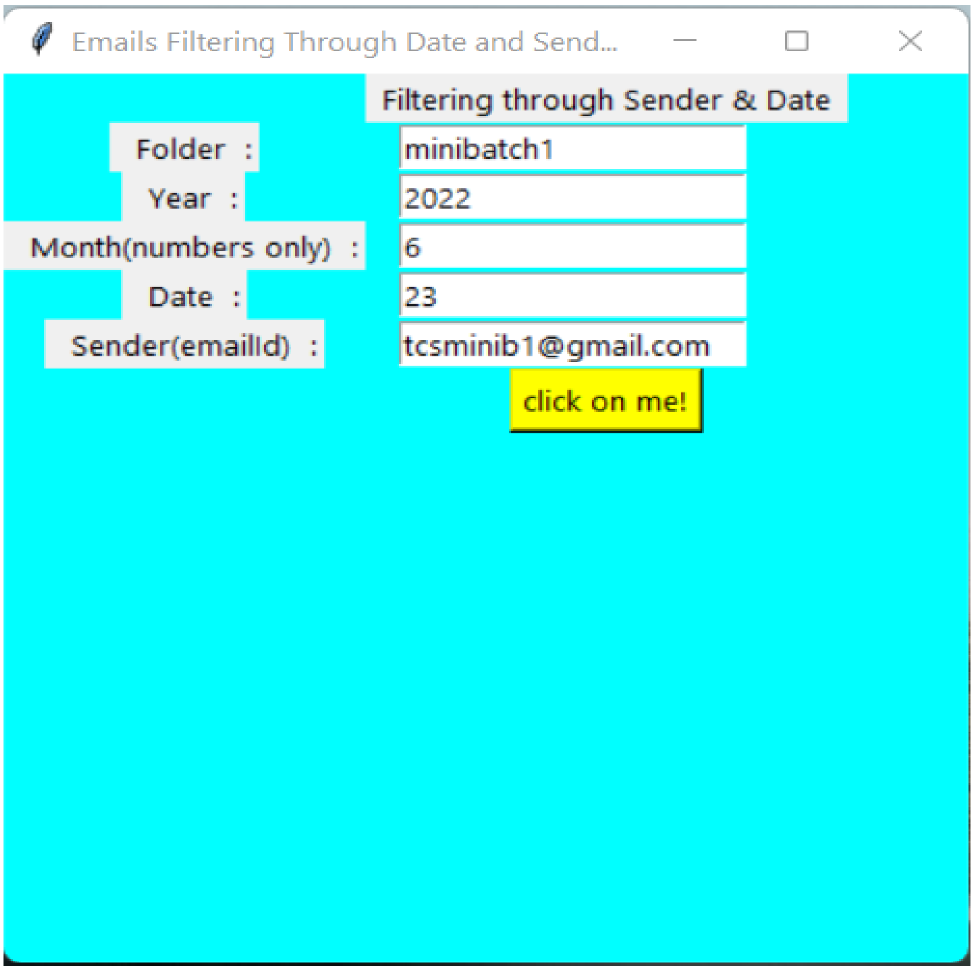
User Name : 1a1236@mvgrce.edu.in

Password : *********

Choosing the Filter Option:



Entering Required Details:

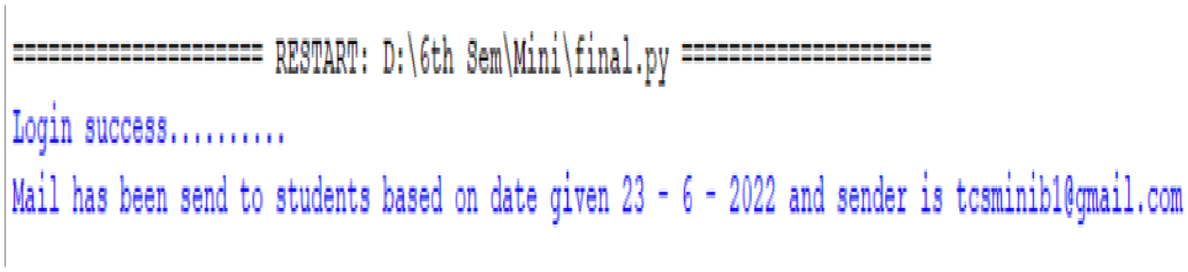


Emails Filtering Through Date and Send...

Filtering through Sender & Date

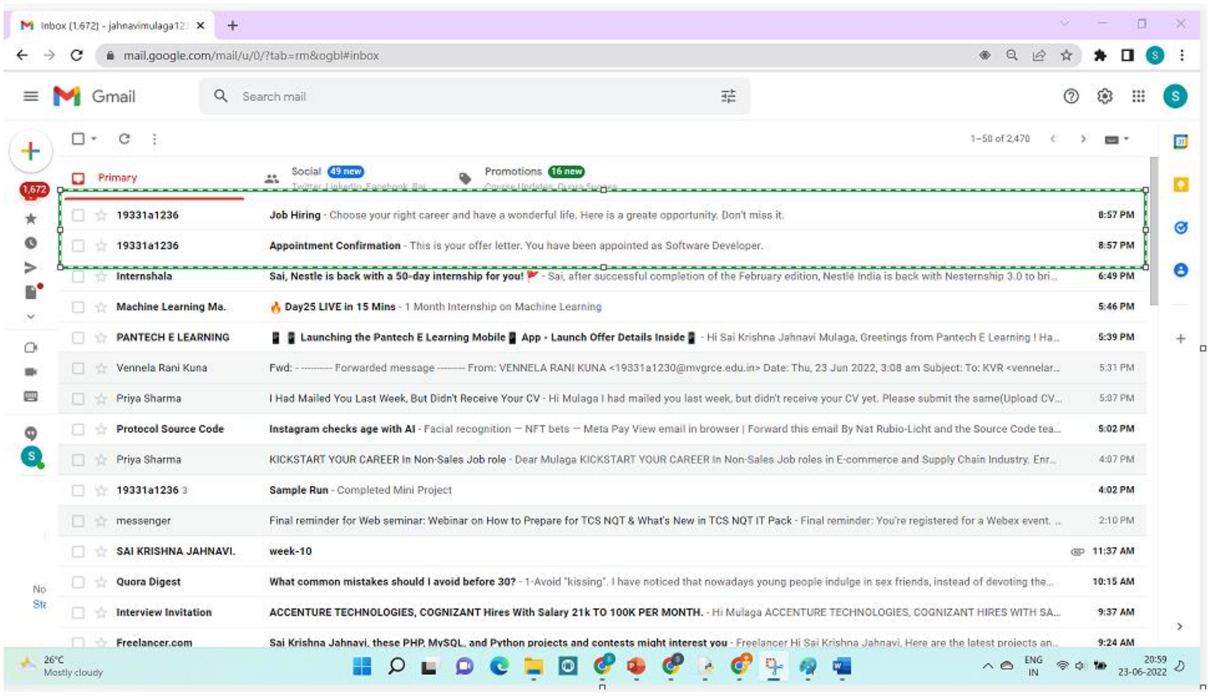
```
Folder  :                  minibatch1
Year  :                    2022
Month(numbers only)  :     6
Date  :                    23
Sender(emailId)  :         tcsminib1@gmail.com
```

Python Console:



```
==================== RESTART: D:\6th Sem\Mini\final.py ====================
Login success..........
Mail has been send to students based on date given 23 - 6 - 2022 and sender is tcsminib1@gmail.com
```

Student mail inbox:



19331a1236	**Job Hiring** - Choose your right career and have a wonderful life. Here is a greate opportunity. Don't miss it.	8:57 PM
19331a1236	**Appointment Confirmation** - This is your offer letter. You have been appointed as Software Developer.	8:57 PM
Internshala	**Sai, Nestle is back with a 50-day internship for you!** 🚩 - Sai, after successful completion of the February edition, Nestlé India is back with Nesternship 3.0 to bri...	6:49 PM
Machine Learning Ma.	🔥 **Day25 LIVE in 15 Mins** - 1 Month Internship on Machine Learning	5:46 PM
PANTECH E LEARNING	**Launching the Pantech E Learning Mobile App - Launch Offer Details Inside** - Hi Sai Krishna Jahnavi Mulaga, Greetings from Pantech E Learning ! Ha...	5:39 PM
Vennela Rani Kuna	Fwd: - ---------- Forwarded message --------- From: VENNELA RANI KUNA <19331a1230@mvgrce.edu.in> Date: Thu, 23 Jun 2022, 3:08 am Subject: To: KVR <vennelar...	5:31 PM
Priya Sharma	I Had Mailed You Last Week, But Didn't Receive Your CV - Hi Mulaga I had mailed you last week, but didn't receive your CV yet. Please submit the same(Upload CV...	5:07 PM
Protocol Source Code	**Instagram checks age with AI** - Facial recognition — NFT bets — Meta Pay View email in browser | Forward this email By Nat Rubio-Licht and the Source Code tea...	5:02 PM
Priya Sharma	KICKSTART YOUR CAREER In Non-Sales Job role - Dear Mulaga KICKSTART YOUR CAREER In Non-Sales Job roles in E-commerce and Supply Chain Industry. Enr...	4:07 PM
19331a1236 3	**Sample Run** - Completed Mini Project	4:02 PM
messenger	Final reminder for Web seminar: Webinar on How to Prepare for TCS NQT & What's New in TCS NQT IT Pack - Final reminder: You're registered for a Webex event. ...	2:10 PM
SAI KRISHNA JAHNAVI.	**week-10**	11:37 AM
Quora Digest	**What common mistakes should I avoid before 30?** - 1-Avoid "kissing". I have noticed that nowadays young people indulge in sex friends, instead of devoting the...	10:15 AM
Interview Invitation	**ACCENTURE TECHNOLOGIES, COGNIZANT Hires With Salary 21k TO 100K PER MONTH.** - Hi Mulaga ACCENTURE TECHNOLOGIES, COGNIZANT HIRES WITH SA...	9:37 AM
Freelancer.com	**Sai Krishna Jahnavi, these PHP, MySQL, and Python projects and contests might interest you** - Freelancer Hi Sai Krishna Jahnavi, Here are the latest projects an...	9:24 AM

CONCLUSION

- Placement Messenger helps us to access the email account with email id and password without any browser.
- We can access mails in email account , we can read those mails and we will be able filter those mails based on date and sender’s email id.
- Using this project, we can transfer those mails automatically to others.
- This Placement Messenger is helpful for Education Institutions to inform most important information without delay.

FUTURE SCOPE:

- We can extend the implementation of this application by using frameworks like Django and Flask.
- Filtering process can also be extended based on the concept of email.
- Access control can also emerged in this application to have security.
- Chat bot and voice bot can also introduced in the application.

REFERENCES

- https://realpython.com/python-send-email/
- https://docs.python.org/3/library/imaplib.html
- https://medium.com/analytics-vidhya/email-extraction-using-python-with-some-filters-233ae451f011
- https://www.tutorialspoint.com/python/python_gui_programming.htm