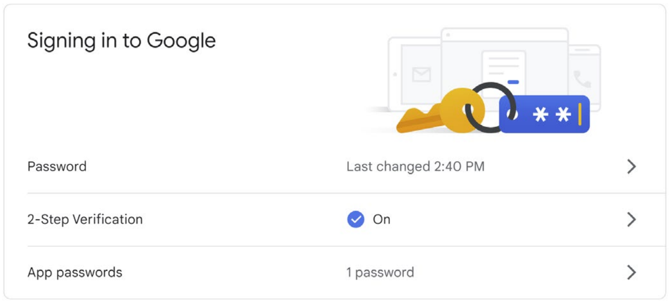
Email programming Python

Gmail Set up

Part 1:

For this project, Python is used to send emails through Gmail. To do this, I needed to use special libraries, including smtp, getpass, and the email.mime module for Python. The specific port is also required; for Gmail, it is 587. Furthermore, Gmail does not allow what they call "less secure apps" to access email accounts by default. So, in order to be able to send email through Python, I had to go into the settings of our test email account and turn on 2-setp-verfication, then set up the "App Password". A password is generated (I will be using it to login).



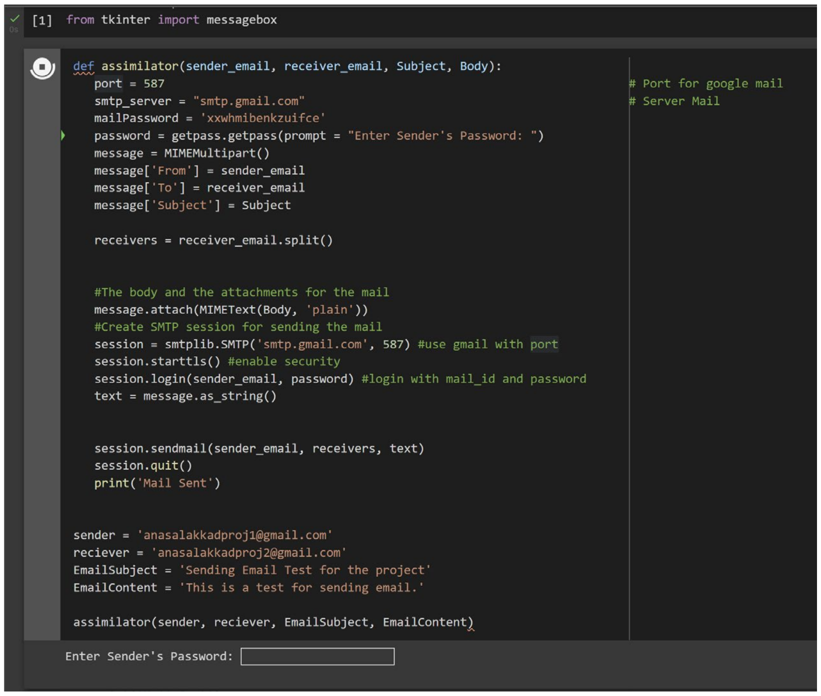
Signing in to Google

Password	Last changed 2:40 PM
2-Step Verification	On
App passwords	1 password

Send email:

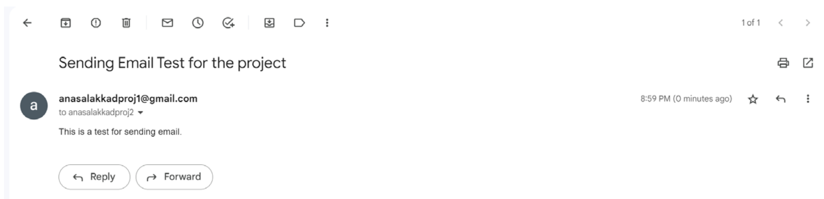
Part 2:

The end product was a function that could easily be called and used to send emails. The function is called "assimilator" and has a couple of inputs for sender of the email, the receiver of the email, subject line of the email, the body of the email, people to send the CC to, and people to send the BCC to. During this portion of the testing, mainly the first four inputs (excluding CC and BCC) were focused on to make sure that they worked before CC and BCC were implemented for later parts.



```
[1] from tkinter import messagebox

def assimilator(sender_email, receiver_email, Subject, Body):
    port = 587                                                          # Port for google mail
    smtp_server = "smtp.gmail.com"                                      # Server Mail
    mailPassword = 'xxwhmibenkzuifce'
    password = getpass.getpass(prompt = "Enter Sender's Password: ")
    message = MIMEMultipart()
    message['From'] = sender_email
    message['To'] = receiver_email
    message['Subject'] = Subject

    receivers = receiver_email.split()


    #The body and the attachments for the mail
    message.attach(MIMEText(Body, 'plain'))
    #Create SMTP session for sending the mail
    session = smtplib.SMTP('smtp.gmail.com', 587) #use gmail with port
    session.starttls() #enable security
    session.login(sender_email, password) #login with mail_id and password
    text = message.as_string()


    session.sendmail(sender_email, receivers, text)
    session.quit()
    print('Mail Sent')


sender = 'anasalakkadproj1@gmail.com'
reciever = 'anasalakkadproj2@gmail.com'
EmailSubject = 'Sending Email Test for the project'
EmailContent = 'This is a test for sending email.'

assimilator(sender, reciever, EmailSubject, EmailContent)

Enter Sender's Password:
```

The below picture is the sender email: anasalakkadproj1@gmail.com

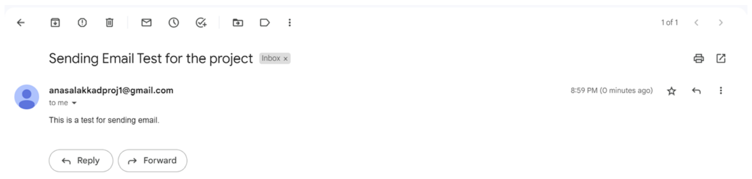


Sending Email Test for the project

anasalakkadproj1@gmail.com
to anasalakkadproj2
8:59 PM (0 minutes ago)

This is a test for sending email.

The below picture is the receiver email: anasalakkadproj2@gmail.com



Sending Email Test for the project

anasalakkadproj1@gmail.com
to me
8:59 PM (0 minutes ago)

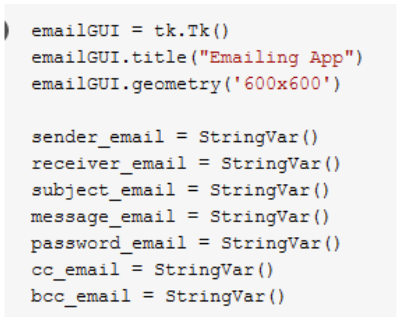
This is a test for sending email.

<u>Part 3:</u> Send email from application

Graphical User Interface (GUI) was designed with all the necessary elements to receive inputs in the application to send the emails. GUI is used in parts 3, 4, and 5.

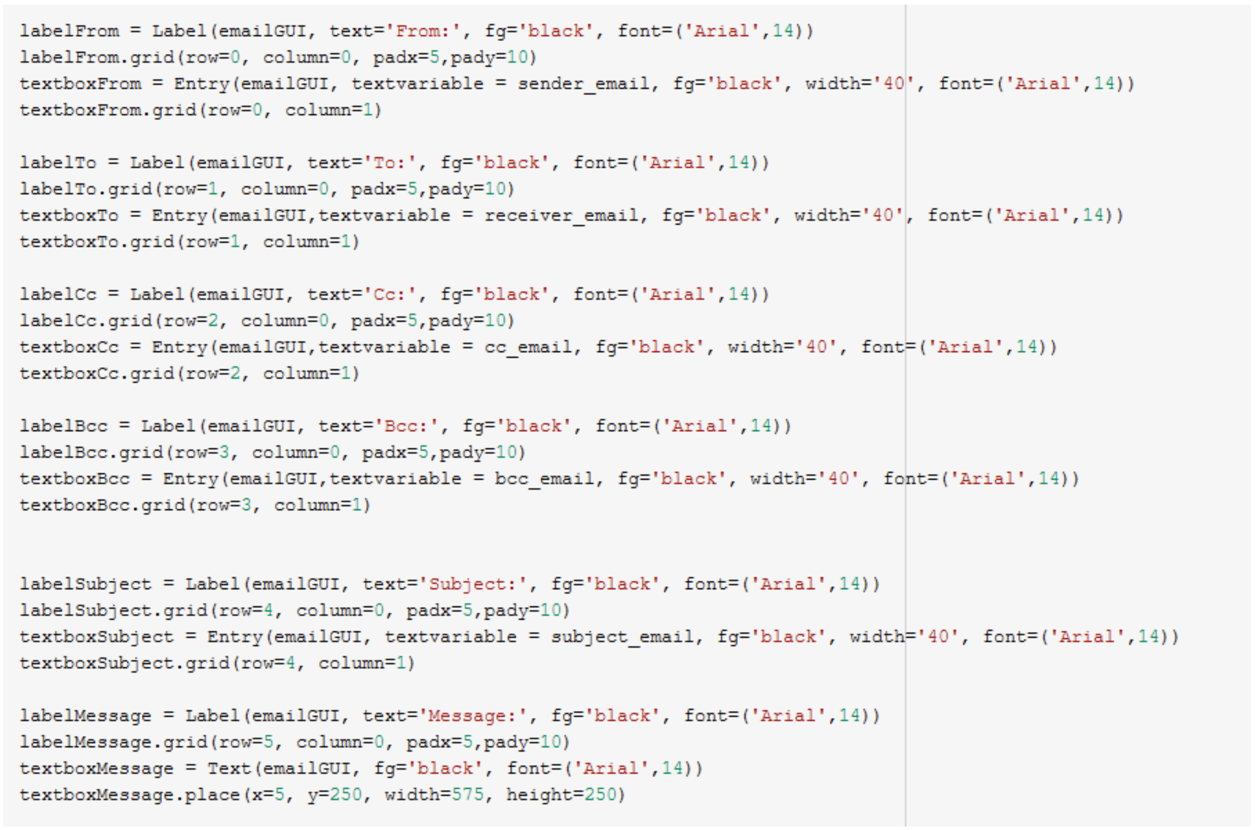
The application allows users to input their email address, the receiving email address, a subject line, and the body of the email. Before sending, it asks the user to input the password for their email account. Then, it sends the email using the same method as part 2. Later, CC and BCC functionality was added for part 4 and 5.

For the application, I used the TK library for GUI. I created a main window and made several strings which the user edits, such as the sender email, receiver, subject line, etc.



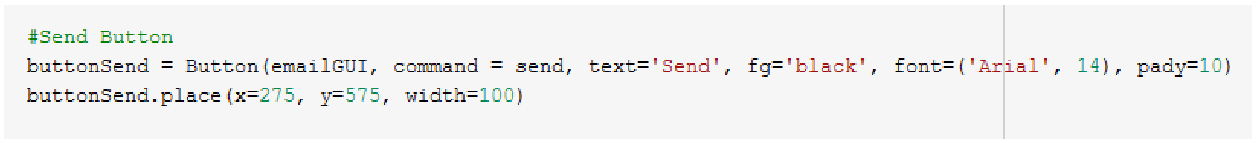
```
emailGUI = tk.Tk()
emailGUI.title("Emailing App")
emailGUI.geometry('600x600')

sender_email = StringVar()
receiver_email = StringVar()
subject_email = StringVar()
message_email = StringVar()
password_email = StringVar()
cc_email = StringVar()
bcc_email = StringVar()
```

Labels are drawn and textboxes which are attached to each string variable are put next to them:

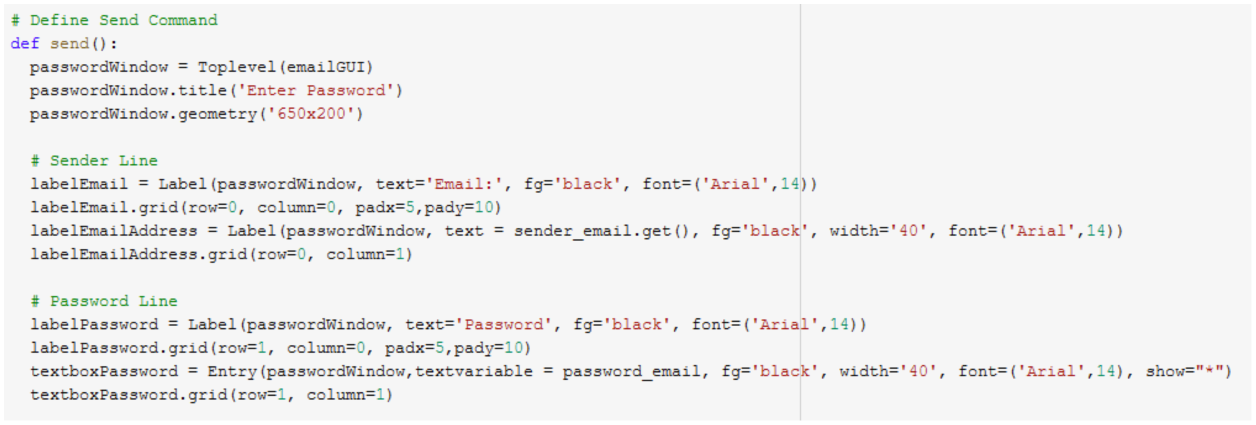


```
labelFrom = Label(emailGUI, text='From:', fg='black', font=('Arial',14))
labelFrom.grid(row=0, column=0, padx=5,pady=10)
textboxFrom = Entry(emailGUI, textvariable = sender_email, fg='black', width='40', font=('Arial',14))
textboxFrom.grid(row=0, column=1)

labelTo = Label(emailGUI, text='To:', fg='black', font=('Arial',14))
labelTo.grid(row=1, column=0, padx=5,pady=10)
textboxTo = Entry(emailGUI,textvariable = receiver_email, fg='black', width='40', font=('Arial',14))
textboxTo.grid(row=1, column=1)

labelCc = Label(emailGUI, text='Cc:', fg='black', font=('Arial',14))
labelCc.grid(row=2, column=0, padx=5,pady=10)
textboxCc = Entry(emailGUI,textvariable = cc_email, fg='black', width='40', font=('Arial',14))
textboxCc.grid(row=2, column=1)

labelBcc = Label(emailGUI, text='Bcc:', fg='black', font=('Arial',14))
labelBcc.grid(row=3, column=0, padx=5,pady=10)
textboxBcc = Entry(emailGUI,textvariable = bcc_email, fg='black', width='40', font=('Arial',14))
textboxBcc.grid(row=3, column=1)


labelSubject = Label(emailGUI, text='Subject:', fg='black', font=('Arial',14))
labelSubject.grid(row=4, column=0, padx=5,pady=10)
textboxSubject = Entry(emailGUI, textvariable = subject_email, fg='black', width='40', font=('Arial',14))
textboxSubject.grid(row=4, column=1)

labelMessage = Label(emailGUI, text='Message:', fg='black', font=('Arial',14))
labelMessage.grid(row=5, column=0, padx=5,pady=10)
textboxMessage = Text(emailGUI, fg='black', font=('Arial',14))
textboxMessage.place(x=5, y=250, width=575, height=250)
```

Once all fields are filled out as desired, a “send” button can be pressed which opens up a new window. The new window displays the sender’s email address and asks for a password to be input.

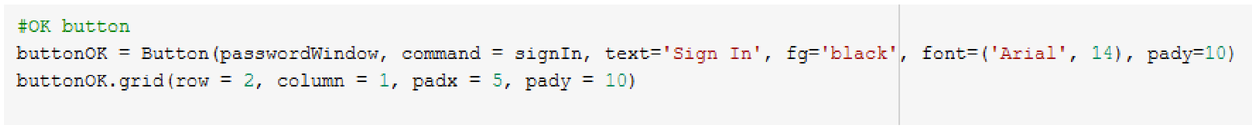


```
#Send Button
buttonSend = Button(emailGUI, command = send, text='Send', fg='black', font=('Arial', 14), pady=10)
buttonSend.place(x=275, y=575, width=100)
```

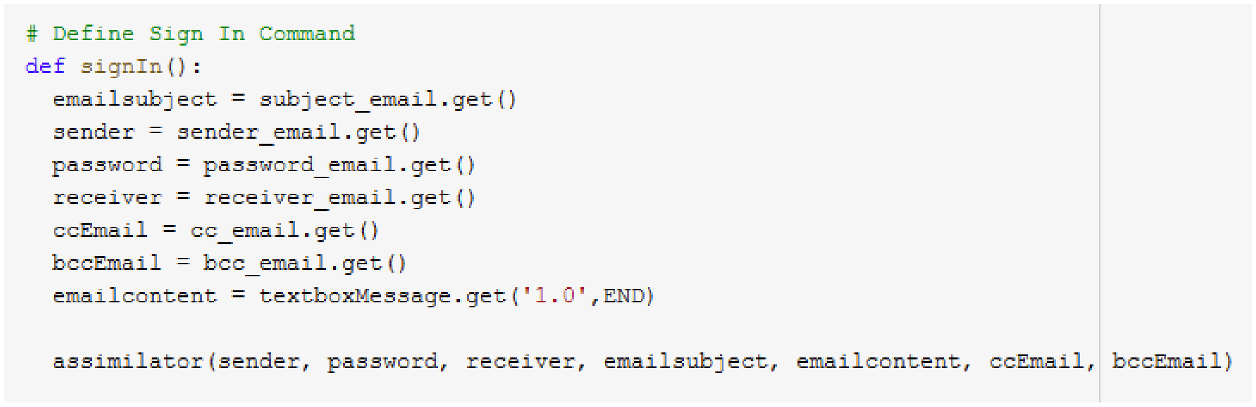


```
# Define Send Command
def send():
  passwordWindow = Toplevel(emailGUI)
  passwordWindow.title('Enter Password')
  passwordWindow.geometry('650x200')

  # Sender Line
  labelEmail = Label(passwordWindow, text='Email:', fg='black', font=('Arial',14))
  labelEmail.grid(row=0, column=0, padx=5,pady=10)
  labelEmailAddress = Label(passwordWindow, text = sender_email.get(), fg='black', width='40', font=('Arial',14))
  labelEmailAddress.grid(row=0, column=1)

  # Password Line
  labelPassword = Label(passwordWindow, text='Password', fg='black', font=('Arial',14))
  labelPassword.grid(row=1, column=0, padx=5,pady=10)
  textboxPassword = Entry(passwordWindow,textvariable = password_email, fg='black', width='40', font=('Arial',14), show="*")
  textboxPassword.grid(row=1, column=1)
```

Once the password is in, a confirmation button can be pressed which will activate the “sign in” command. This command takes all the inputs that the user typed and passes them into the assimilator function designed for part 2.

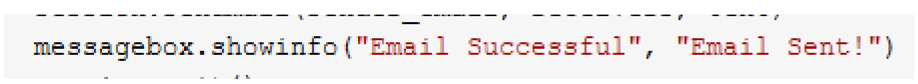


```
#OK button
buttonOK = Button(passwordWindow, command = signIn, text='Sign In', fg='black', font=('Arial', 14), pady=10)
buttonOK.grid(row = 2, column = 1, padx = 5, pady = 10)
```



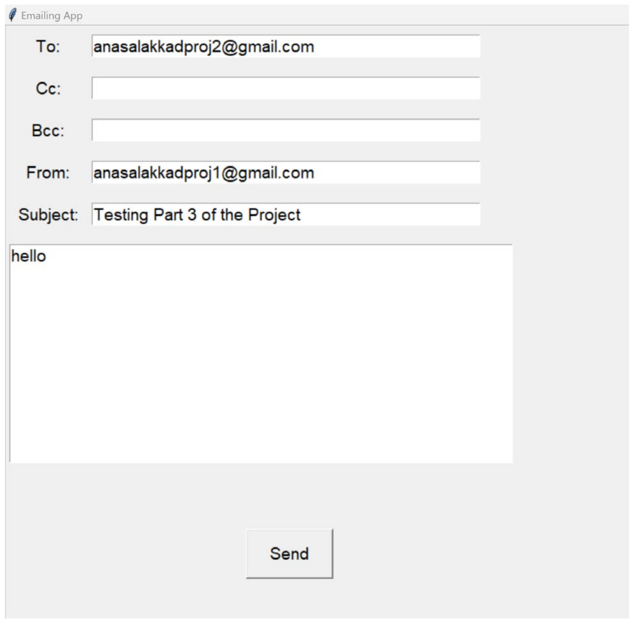
```
# Define Sign In Command
def signIn():
  emailsubject = subject_email.get()
  sender = sender_email.get()
  password = password_email.get()
  receiver = receiver_email.get()
  ccEmail = cc_email.get()
  bccEmail = bcc_email.get()
  emailcontent = textboxMessage.get('1.0',END)

  assimilator(sender, password, receiver, emailsubject, emailcontent, ccEmail, bccEmail)
```

To polish off the application, a message box saying “Email sent!” is added to the assimilator function once its other processes complete.



```
messagebox.showinfo("Email Successful", "Email Sent!")
```

Here is the main window of the application. I can input whatever I want for the different fields:



To: anasalakkadproj2@gmail.com

Cc:

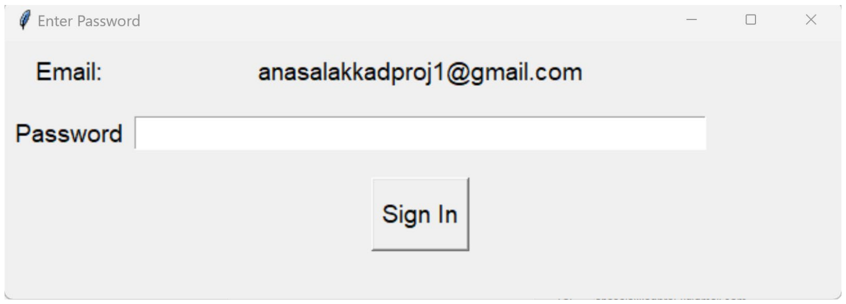
Bcc:

From: anasalakkadproj1@gmail.com

Subject: Testing Part 3 of the Project

hello

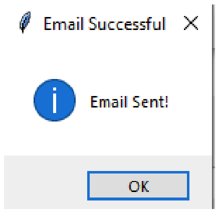
The password window:



Email: anasalakkadproj1@gmail.com

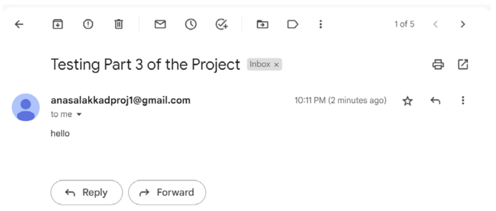
Password

The small pop-up confirming that the email sent:



Email Sent!

And finally, the received email:



Testing Part 3 of the Project

anasalakkadproj1@gmail.com
to me
10:11 PM (2 minutes ago)

hello

<u>CC & BCC</u>

<u>Part 4:</u>

CC stands for carbon copy and when it pertains to emails, CC serves as a list that has the purpose of allowing all recipients to see the other users that are on the CC list.

For this part, I mainly focused on adding the CC element of an email, and checked the functionality of the feature. This feature was simple as I could just add a line for “cc” and append the list of emails sent to include the people on CC.

GUI changes: To add the CC to the application, I had to ensure that there was a corresponding string variable, and a label with text box for the user to input the desired CC addresses.

```
cc_email = StringVar()
```

```
labelCc = Label(emailGUI, text='Cc:', fg='black', font=('Arial',14))
labelCc.grid(row=2, column=0, padx=5,pady=10)
textboxCc = Entry(emailGUI,textvariable = cc_email, fg='black', width='40', font=('Arial',14))
textboxCc.grid(row=2, column=1)
```

Then, CC is read from the text box and passed into the modified assimilator function

```
ccEmail = cc_email.get()
```

```
assimilator(sender, password, receiver, emailsubject, emailcontent, ccEmail, bccEmail)
```

Label and text box:

<u>Part 5:</u>

BCC stand for blind carbon copy, and it is another list of email recipients, but unlike CC no users (apart from the sender) can see the people on this list. If I were to put the BCC list into the “Bcc” function of sending, it would be visible to the other users. To implement BCC, I just simply added the users to the list of email recipients without adding them to a function (which would make them visible to other users). Next, I added the necessary portions to the GUI portion of the application to work properly.

GUI changes: BCC is added to the CC in essentially the same way as CC. A new string variable is added and a label with textbox is drawn.

```
bcc_email = StringVar()
```

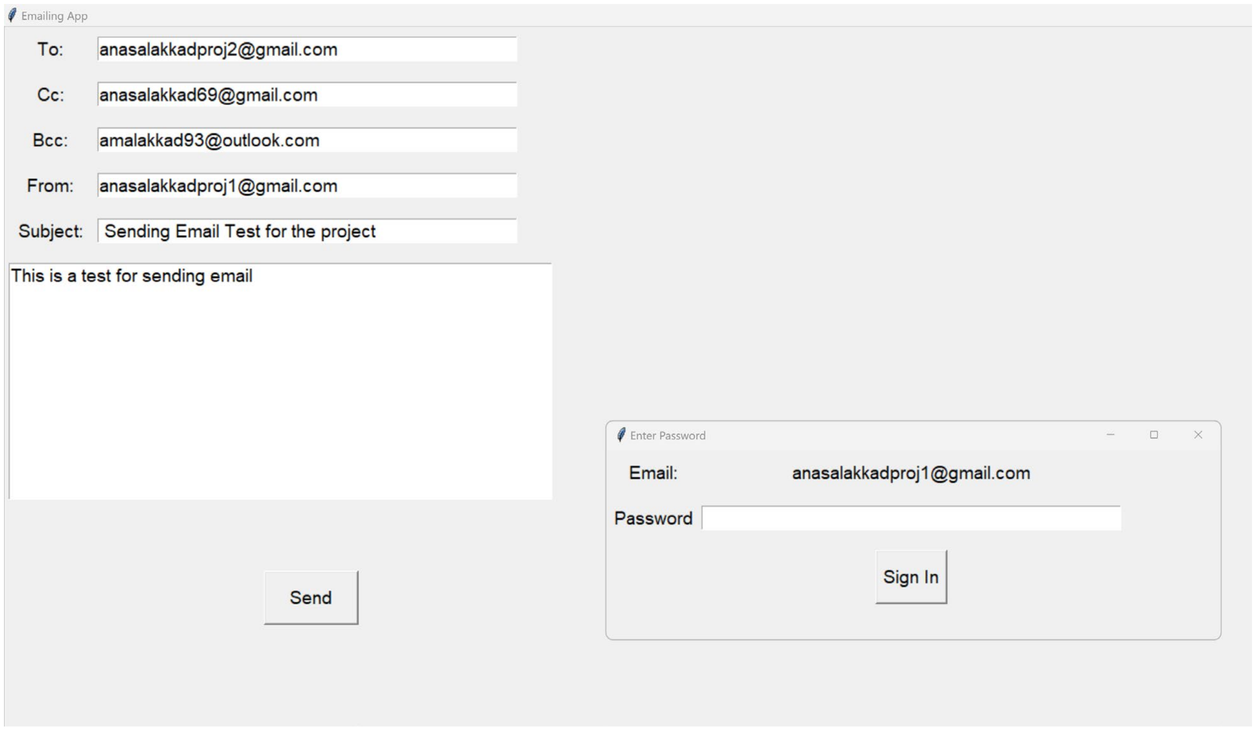
```
labelBcc = Label(emailGUI, text='Bcc:', fg='black', font=('Arial',14))
labelBcc.grid(row=3, column=0, padx=5,pady=10)
textboxBcc = Entry(emailGUI,textvariable = bcc_email, fg='black', width='40', font=('Arial',14))
textboxBcc.grid(row=3, column=1)
```

When the user wants to send the email, the BCC input is read and passed into the assimilator function

```
bccEmail = bcc_email.get()
```

Label and text box:

Below is the full picture of the output:



To: anasalakkadproj2@gmail.com

Cc: anasalakkad69@gmail.com

Bcc: amalakkad93@outlook.com

From: anasalakkadproj1@gmail.com

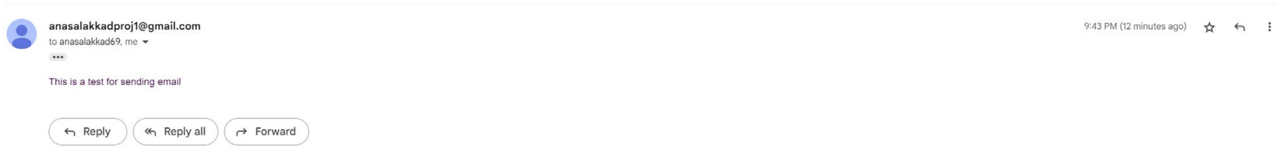
Subject: Sending Email Test for the project

This is a test for sending email

Email: anasalakkadproj1@gmail.com

Password

Below is the receiver Email: anasalakkadproj2@gmail.com

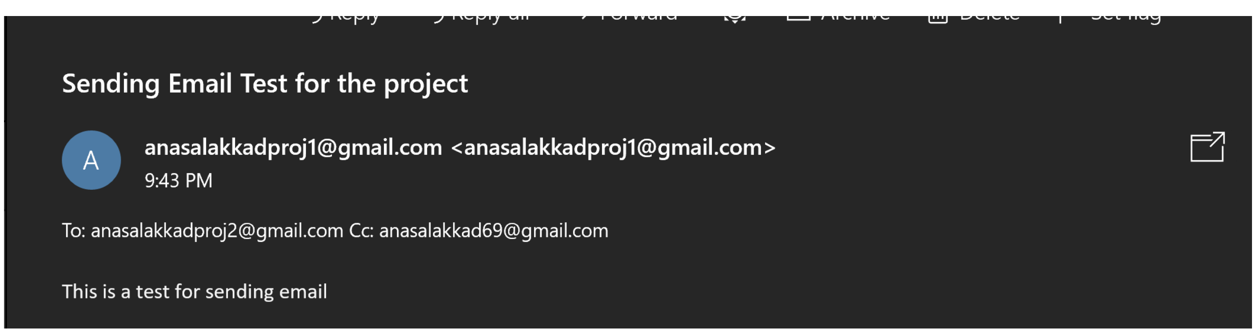

anasalakkadproj1@gmail.com

to anasalakkad69, me

9:43 PM (12 minutes ago)

This is a test for sending email

Below is the CC Email: anasalakkad69@gmail.com


Sending Email Test for the project

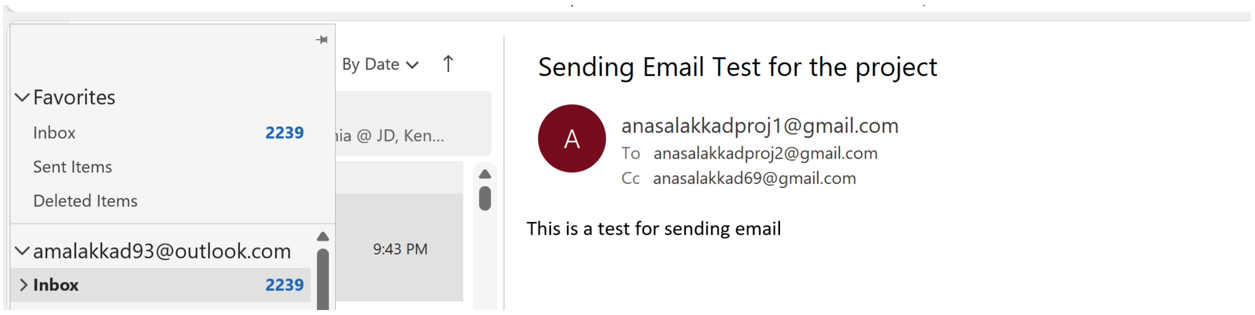
anasalakkadproj1@gmail.com <anasalakkadproj1@gmail.com>

9:43 PM

To: anasalakkadproj2@gmail.com Cc: anasalakkad69@gmail.com

This is a test for sending email

Below is the BCC Email: amalakkad93@outlook.com



Sending Email Test for the project

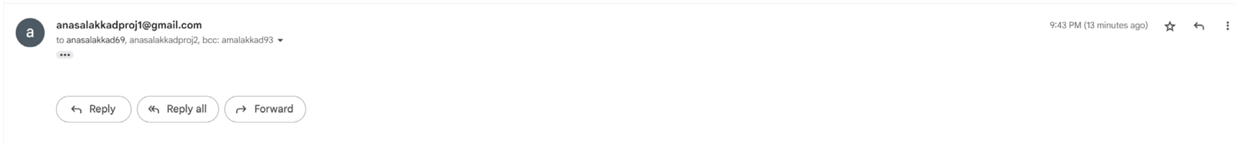
anasalakkadproj1@gmail.com

To anasalakkadproj2@gmail.com

Cc anasalakkad69@gmail.com

This is a test for sending email

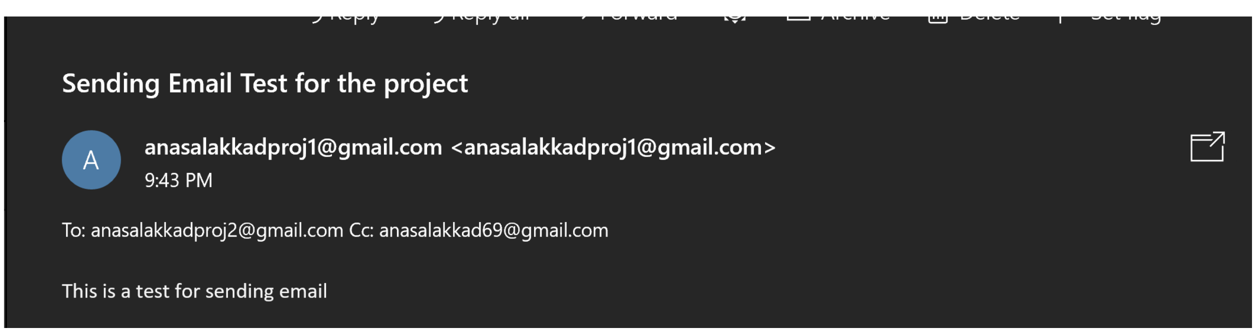
Below is the sender Email: anasalakkadproj1@gmail.com


anasalakkadproj1@gmail.com

to anasalakkad69, anasalakkadproj2, bcc: amalakkad93

9:43 PM (13 minutes ago)

Below it shows all the emails except amalakkad93@outlook.com.

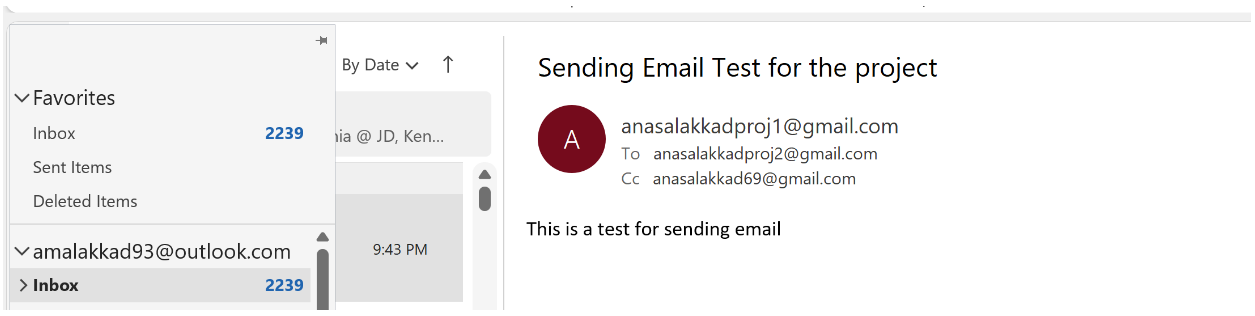

Sending Email Test for the project

anasalakkadproj1@gmail.com <anasalakkadproj1@gmail.com>

9:43 PM

To: anasalakkadproj2@gmail.com Cc: anasalakkad69@gmail.com

This is a test for sending email



Sending Email Test for the project

anasalakkadproj1@gmail.com
To anasalakkadproj2@gmail.com
Cc anasalakkad69@gmail.com

This is a test for sending email

From the screenshots above we can see that all the emails can see which email received the message “This is a test for sending email”. The only email was invisible is the BCC Email: amalakkad93@outlook.com