E-mail Implementation Using SMTP Protocol

A Project Report

Department of Computer Application

National Institute of Technology Tiruchirappalli

Submitted by:

SHIVAM SINGH

205120098

Submitted to:

Dr. T RUSO

INTRODUCTION

The project is an online platform developed for user to make send mail for their personal use.

It facilitates those user who don't want to open browser and then open gmail and then send there mail , time related issues and other issues. It provides services in very less time and in hassel free manner that will not only save their time but also do their task in organized and professional way .

In our report, we first discuss the related literature which covers the details about the project and the existing approaches. Subsequently, we discuss the methodology used in our approach. Later, we discuss the application of this model to achieve the aforementioned objectives. Finally, we conclude the report along with advantages, limitations and future works by presenting an analysis of results obtained.

Goals of the proposed system

1. **Planned approach towards working: -** The working in the organization is well planned and organized. The data will be stored properly in data stores, which will help in retrieval of information as well as its storage.

2. **Accuracy: -** The level of accuracy in the proposed system will be higher. All operation would be done correctly and it ensures that whatever information is coming from the center is accurate.

3. **Reliability:** - The reliability of the proposed system will be high due to the above stated reasons. The reason for the increased reliability of the system is that now there would be proper storage of information.

4. **No Redundancy: -** In the proposed system utmost care would be that no information is repeated anywhere, in storage or otherwise. This would assure economic use of storage space and consistency in the data stored.

5. **Immediate retrieval of information: -** The main objective of proposed system is to provide for a quick and efficient retrieval of information.

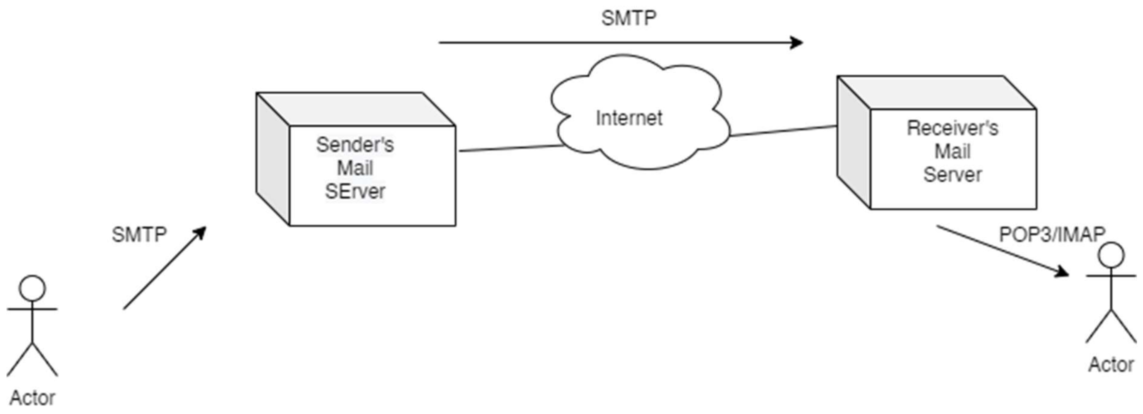
6. **Easy to Operate: -** The system should be easy to operate and should be such that it can be developed within a short period of time and fit in the limited budget of the user.

Language and Library Used:

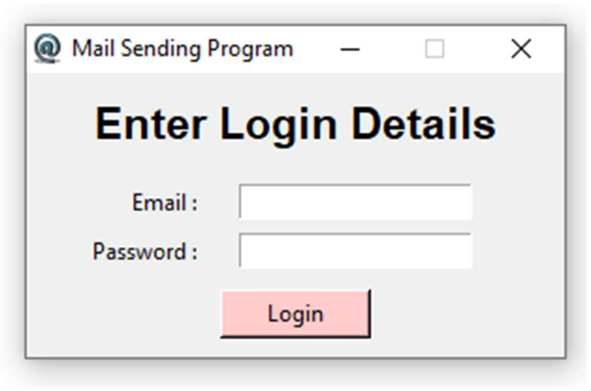
We use PYTHON language in for this mini project and some libraries:-

- **Tkinter**
- **Smtplib**
- **Re**

Project Diagram :



Project synopsis:


Mail Sending Program

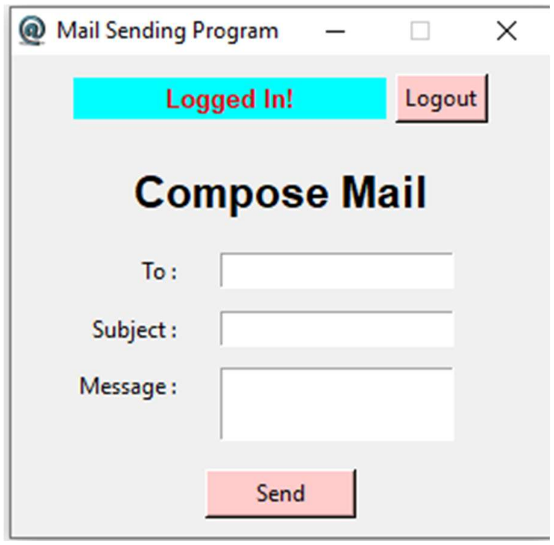
Enter Login Details

Email :

Password :

Login

This is login screen.


Mail Sending Program

Logged In! Logout

Compose Mail

To :

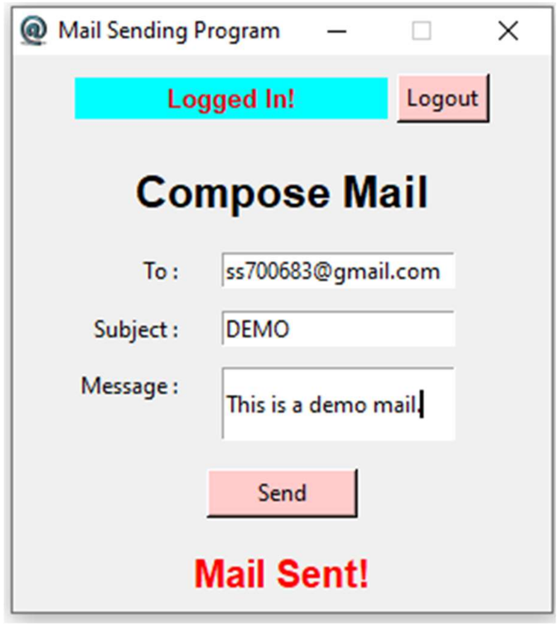
Subject :

Message :

Send

This screen will appear when you

Successfully logged in.


Mail Sending Program

Logged In! Logout

Compose Mail

To : ss700683@gmail.com

Subject : DEMO

Message : This is a demo mail

Send

Mail Sent!

This screen appears after

Sending mail.

Future work in project

- We will add more functionality in our project
- We will be able to attach pic or document file in future
- We will schedule our mail
- We will able to send mail more than one person.

Project Code:

```
from tkinter import *

import smtplib

import re




def start_logging():


    if login_validation():


        global username

        username = str(e1.get())

        password = str(e2.get())

        try:

            global server
            server = smtplib.SMTP('smtp.gmail.com:587')

            server.ehlo()

            server.starttls()

            server.login(username, password)

            fm2.pack()

            b3.grid()

            lbl4['text'] = "Logged In!"

            root.after(10, root.grid)

            fm.pack_forget()

            root.after(10, root.grid)

            fm3.pack()

            lbl9.grid_remove()

            root.after(10, root.grid)


        except Exception as e:

            fm2.pack()

            lbl4.grid()

            lbl4['text'] = "Error in Login!"

            b3.grid_remove()

            root.after(10, root.grid)



def hide_login_label():
    fm2.pack_forget()

    fm3.pack_forget()

    root.after(10, root.grid)




def send_mail():


    if msg_validation():


        lbl9.grid_remove()

        root.after(10, root.grid)

        receiver = str(e3.get())

        subject = str(e4.get())

        msgbody = str(e5.get())


        msg = "From: " + username + "\n" + "To: " + receiver + \

            "\n" + "Subject: " + subject + "\n" + msgbody


        try:

            server.sendmail(username, receiver, msg)

            lbl9.grid()

            lbl9['text'] = "Mail Sent!"

            root.after(10, lbl9.grid)
        except Exception as e:
            lbl9.grid()
            lbl9['text'] = "Error in Sending Mail!"
            root.after(10, lbl9.grid)


def logout():
    try:
        server.quit()
        fm3.pack_forget()
        fm2.pack()
        lbl4.grid()
        lbl4['text'] = "Logged out successfully!"
        b3.grid_remove()
        fm.pack()
        e2.delete(0, END)
        root.after(10, root.grid)
    except Exception as e:
        lbl4['text'] = "Error in Logout!"


def login_validation():
    email_text = str(e1.get())
    pass_text = str(e2.get())

    if (email_text == "") or (pass_text == ""):

        fm2.pack()

        lbl4.grid()

        lbl4['text'] = "Fill all the Places!"

        b3.grid_remove()

        root.after(10, root.grid)

        return False

    else:

        EMAIL_REGEX = re.compile(r"[^@\s]+@[^@\s]+\.[a-zA-Z0-9]+$")

        if not EMAIL_REGEX.match(email_text):

            fm2.pack()

            lbl4.grid()

            lbl4['text'] = "Enter a valid Email!"

            b3.grid_remove()

            root.after(10, root.grid)

            return False

        else:

            return True




def msg_validation():

    email_text = str(e3.get())
```

```
sub_text = str(e4.get())

msg_text = str(e5.get())

if (email_text == "") or (sub_text == "") or (msg_text == ""):

    lbl9.grid()

    lbl9['text'] = "Fill all the Places!"

    root.after(10, root.grid)

    return False

else:

    EMAIL_REGEX = re.compile(r"[^@\s]+@[^@\s]+\.[a-zA-Z0-9]+$")

    if not EMAIL_REGEX.match(email_text):

        lbl9.grid()

        lbl9['text'] = "Enter a valid Email!"

        root.after(10, root.grid)

        return False

    elif (len(sub_text) < 3) or (len(msg_text) < 3):

        lbl9.grid()

        lbl9['text'] = "Enter atleast 3 character!"

        root.after(10, root.grid)

        return False

    else:

        return True
```

```
root = Tk()

root.title('Mail Sending Program')

root.resizable(False, False)


root.iconbitmap("mail.ico")




fm = Frame(root, width=1200, height=600)

fm.pack(side=TOP, expand=NO, fill=NONE)




lbl1 = Label(fm, width=20, text="Enter Login Details",

            font=("Helvetica 17 bold"))

lbl1.grid(row=0, columnspan=3, pady=10)


lbl2 = Label(fm, text="Email : ").grid(row=1, sticky=E, pady=5)

lbl3 = Label(fm, text="Password : ").grid(row=2, sticky=E)


e1 = Entry(fm)

e2 = Entry(fm, show="*")


e1.grid(row=1, column=1, pady=5)

e2.grid(row=2, column=1)
```

```
b1 = Button(fm, text="Login", width=10, bg="#ffcccc",
        fg="black", command=lambda: start_logging())
b1.grid(row=3, columnspan=3, pady=10)

fm2 = Frame(root)
fm2.pack(side=TOP, expand=NO, fill=NONE)

lbl4 = Label(fm2, width=20, bg="cyan", fg="red",
         text="Logged In!", font=("Helvetica 10 bold"))
lbl4.grid(row=0, column=0, columnspan=2, pady=5)

b3 = Button(fm2, text="Logout", bg="#ffcccc",
        fg="black", command=lambda: logout())
b3.grid(row=0, column=4, sticky=E, pady=10, padx=(5, 0))


fm3 = Frame(master=root)
fm3.pack(side=TOP, expand=NO, fill=NONE)

lbl5 = Label(fm3, width=20, text="Compose Mail", font=("Helvetica 17 bold"))
lbl5.grid(row=0, columnspan=3, pady=10)
```

```
lbl6 = Label(fm3, text="To : ").grid(row=1, sticky=E, pady=5)

lbl7 = Label(fm3, text="Subject : ").grid(row=2, sticky=E, pady=5)

lbl8 = Label(fm3, text="Message : ").grid(row=3, sticky=E)


e3 = Entry(fm3)

e4 = Entry(fm3)

e5 = Entry(fm3)


e3.grid(row=1, column=1, pady=5)

e4.grid(row=2, column=1, pady=5)

e5.grid(row=3, column=1, pady=5, rowspan=3, ipady=10)


b2 = Button(fm3, text="Send", width=10, bg="#ffcccc",

        fg="black", command=lambda: send_mail())

b2.grid(row=6, columnspan=3, pady=10)


lbl9 = Label(fm3, width=20, fg="red", font=("Helvetica 15 bold"))

lbl9.grid(row=7, columnspan=3, pady=5)


hide_login_label()


root.mainloop()
```