In [ ]:

Send Automatic Emails using Python

Every time you register on a new app, you automatically receive a welcome message with your name on it. If you want to learn how to send such emails automatically, then this article is for you. In this article, I will walk you through how to send automatic emails using Python.

In [ ]:

How to Send Automatic Emails using Python?

To send automatic emails using Python, you must first understand how to send an email using the Python programming language. Once you know how to send an email using Python, the next thing you need to figure out is what you want to send automatically. For example, many companies send OTP or welcome messages, while some companies even send newsletters to newly registered users.

So in the section below, I will take you through how to send automatic emails using Python. I will automatically send a welcome message to the newly registered user. For this task, you must first generate a google app password for your Gmail account. If you don’t know how to generate it, you can learn more from the video below.

In [ ]:

OTP Verification using Python

In [ ]:

Do you know how you get a unique OTP every time you go through the payment process in an online transaction? Each company has its ways of creating an OTP for verification, but most of the companies have their systems programmed to generate a 6-digit random number. In this article, I will walk you through the task of OTP verification using Python. By the end of this article, you will be able to send a unique OTP to any email id for the OTP verification task.

In [ ]:

Steps to Create an OTP Verification System using Python

OTP Verification is the process of verifying a user by sending a unique password so that the user can be verified before completing a registration or payment process. Most of the time, we get an OTP when we make an online payment, or when we forget our password, or when creating an account on any online platform. Thus, the sole purpose of an OTP is to verify the identity of a user by sending a unique password.

1. First, create a 6-digit random number
2. Then store the number in a variable
3. Then we need to write a program to send emails
4. When sending email, we need to use OTP as a message
5. Finally, we need to request two user inputs; first for the user's email and then for the OTP that the user has received.

So this is the complete process of creating an OTP verification application using Python. In the section below, I will take you through how to implement these steps using Python for the task of OTP verification.

I hope you now have understood what is an OTP and how we can create an application for the task of OTP verification. Now let's follow the steps mentioned above by using Python to create an application for the task of OTP verification. I will start by importing the necessary Python library that we need for this task:

In [ ]:

```

```

In [7]:

```
import os
import math
import random
import smtplib
```

Now I will generate a random number and store it in a variable which I will be using while sending emails to the users:

In [8]:

```
digits="7989239822"
OTP="123456"
for i in range(6):
    OTP+=digits[math.floor(random.random()*10)]
otp = OTP + " is your OTP"
msg= otp
```

Now, before we go ahead, you need to have your Google app password to be able to send emails using your Gmail account. For this task, you need to follow the steps mentioned here. After you create your app password for your Gmail account you will get a key. Copy that key and paste in the code below to send emails for OTP verification using Python:

In [9]:

```
s = smtplib.SMTP('smtp.gmail.com', 587)
s.starttls()
s.login("Your Gmail Account", "You app password")
emailid = input("Enter your email: ")
s.sendmail('&&&&&&&&&&&',emailid,msg)
a = input("Enter Your OTP >>: ")
if a == OTP:
    print("Verified")
else:
    print("Please Check your OTP again")
```

```
---------------------------------------------------------------------------
SMTPAuthenticationError                   Traceback (most recent call last)
Cell In [9], line 3
      1 s = smtplib.SMTP('smtp.gmail.com', 587)
      2 s.starttls()
----> 3 s.login("Your Gmail Account", "You app password")
      4 emailid = input("Enter your email: ")
      5 s.sendmail('&&&&&&&&&&&',emailid,msg)

File /opt/conda/lib/python3.10/smtplib.py:750, in SMTP.login(self, user, password, in
itial_response_ok)
    747         last_exception = e
    749 # We could not login successfully.  Return result of last attempt.
--> 750 raise last_exception

File /opt/conda/lib/python3.10/smtplib.py:739, in SMTP.login(self, user, password, in
itial_response_ok)
    737 method_name = 'auth_' + authmethod.lower().replace('-', '_')
    738 try:
--> 739     (code, resp) = self.auth(
    740         authmethod, getattr(self, method_name),
    741         initial_response_ok=initial_response_ok)
    742     # 235 == 'Authentication successful'
    743     # 503 == 'Error: already authenticated'
    744     if code in (235, 503):

File /opt/conda/lib/python3.10/smtplib.py:662, in SMTP.auth(self, mechanism, authobje
ct, initial_response_ok)
    660 if code in (235, 503):
    661     return (code, resp)
--> 662 raise SMTPAuthenticationError(code, resp)

SMTPAuthenticationError: (535, b'5.7.8 Username and Password not accepted. Learn more
at\n5.7.8  https://support.google.com/mail/?p=BadCredentials s1-20020a056a00178100b00
625d84a0194sm6976394pfg.107 - gsmtp')
```

Once you run this code you enter an email where you want to send an OTP and then enter the OTP that you have received in the email. You can get the complete code used in this article for the task of OTP verification from below.

In [10]:
```
import os
import math
import random
import smtplib

digits="0123456789"
OTP=""
for i in range(6):
    OTP+=digits[math.floor(random.random()*10)]
otp = OTP + " is your OTP"
msg= otp
s = smtplib.SMTP('smtp.gmail.com', 587)
```

```
s.starttls()
s.login("Your Gmail Account", "You app password")
emailid = input("Enter your email: ")
s.sendmail('&&&&&&&&&&&',emailid,msg)
a = input("Enter Your OTP >>: ")
if a == OTP:
    print("Verified")
else:
    print("Please Check your OTP again")
```

```
---------------------------------------------------------------------------
SMTPAuthenticationError                   Traceback (most recent call last)
Cell In [10], line 14
     12 s = smtplib.SMTP('smtp.gmail.com', 587)
     13 s.starttls()
---> 14 s.login("Your Gmail Account", "You app password")
     15 emailid = input("Enter your email: ")
     16 s.sendmail('&&&&&&&&&&&',emailid,msg)

File /opt/conda/lib/python3.10/smtplib.py:750, in SMTP.login(self, user, password, in
itial_response_ok)
    747         last_exception = e
    749 # We could not login successfully.  Return result of last attempt.
--> 750 raise last_exception

File /opt/conda/lib/python3.10/smtplib.py:739, in SMTP.login(self, user, password, in
itial_response_ok)
    737 method_name = 'auth_' + authmethod.lower().replace('-', '_')
    738 try:
--> 739     (code, resp) = self.auth(
    740         authmethod, getattr(self, method_name),
    741         initial_response_ok=initial_response_ok)
    742     # 235 == 'Authentication successful'
    743     # 503 == 'Error: already authenticated'
    744     if code in (235, 503):

File /opt/conda/lib/python3.10/smtplib.py:662, in SMTP.auth(self, mechanism, authobje
ct, initial_response_ok)
    660 if code in (235, 503):
    661     return (code, resp)
--> 662 raise SMTPAuthenticationError(code, resp)

SMTPAuthenticationError: (535, b'5.7.8 Username and Password not accepted. Learn more
at\n5.7.8  https://support.google.com/mail/?p=BadCredentials j9-20020a056a00234900b00
63d3d776910sm6930580pfj.138 - gsmtp')
```

```
In [ ]:
```

Send Automatic Emails using Python

It is very important to generate a Google app password for your Gmail account, as you will be sending automatic emails using Python through your Gmail account. Once you've generated your Google app password, here's how you can start the task of sending emails using Python:

In [ ]:
```
import os
import random
import smtplib


def automatic_email():
    user = input("Enter Your Name >>: ")
    email = input("Enter Your Email >>: ")
    message = (f"Dear {user}, Welcome to Thecleverprogrammer")
    s = smtplib.SMTP('smtp.gmail.com', 587)
    s.starttls()
    s.login("Your Gmail Account", "Your App Password")
    s.sendmail('&&&&&&&&&&&', email, message)
    print("Email Sent!")

automatic_email()
```

In the code above I have defined a Python function that will automatically send emails to a newly registered user. This function will start by asking for the user's name and email. Then the user's name will be stored in the message. Then in the 12th line of the code above, you need to replace the first parameter with your email and the second parameter with the google app password you generated before. After that just run the above code and you will see the output as shown below.

In [ ]:

After filling in details to these user inputs I got an email in my inbox as shown below.

Summary

So this is how you can send automatic emails using Python. To send automatic emails using Python, you must first understand how to send an email using the Python programming language. Once you know how to send an email using Python, the next thing you need to figure out is what you want to send automatically. I hope you liked this articles on how to send emails automatically using Python. Feel free to ask your valuable questions in the comments section below.

In [ ]: