PYTHON PROJECT CODE: -

TOPIC – OTP GENERATOR

```
import math
import random
import smtplib

str = "0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"
OTP = ""
varlen= len(str)
# Now we will generate a random number and store it in a variable
## which we will be using while sending emails to the users:
for i in range(6):
    OTP += str[math.floor(random.random() * varlen)]
#random.random returns a random float in the range [0.0, 1.0)
## *10 to keep the otp in range of 0 to 9
### math. floor() function returns the largest integer not greater than x
#Python already has a library that lets you connect to an SMTP server, like the one Gmail uses. This library is called, predictably, smtplib and comes included with Python.

#SMTP (Simple Mail Transfer Protocol) is an application-level protocol (on top of TCP) used to communicate with mail servers from external services, like an email client on your phone. SMTP is a delivery protocol only, so you can't actually retrieve email with it, you can only send email, which is what we'll be focusing on in this article. If you want to retrieve email instead, then you'll want to check out the IMAP (Internet Message Access Protocol) protocol.
```

#I should note that many email services, like Gmail, don't usually use SMTP on their internal mail servers. SMTP is usually just provided as an outward-facing interface to their service via the smtp.gmail.com server. This is mostly meant to be used by email clients on your phone or computer (like Outlook, Thunderbird, etc).

#Using smtplib library, there are a few different ways you can create a connection to your mail server. In this section, we'll focus on creating a plain, insecure connection (which should rarely, if ever, be used). This connection is unencrypted and defaults to port 25. However, the protocol for mail submission actually uses 587, which is what we'll use.

#starttls() is an email protocol command that tells an email server that an email client, including an email client running in a web browser, wants to turn an existing insecure connection into a secure one.

```
otp = OTP + " is your OTP"
with smtplib.SMTP("smtp.gmail.com") as s:
    s.starttls()
#encrypiton
    emailid = input("Enter your email: ")
    s.login(user="pavanramina@gmail.com", password="fxjnfaofgsqbsfrr")
    s.sendmail(
        from_addr="pavanramina@gmail.com",
        to_addrs=emailid,
        msg="Subject: Your OTP\n\n" + otp,
    )


a = input("Enter your OTP >>: ")
if a == OTP:
    print("Verified")
else:
    print("Please Check your OTP again")
```