Documentation of Cap Stone Project

(OTP Verification System)

Name: Ch.Rama Sai pramodh

Student\_ID:S\_8599

***Modules used in project:*** In project random, AppJar these modules are used.

**Coding language used:** Python.

***IDE Used:*** PyCharm, Google Colab.

***Code breakdown and Explanation:***

***Step1: OTP Generation:***

For generating the OTP(one time password) random module is used and the randint is used to generate 6-digit OTP.

# # for generating the OTP  
import random  
def generated\_otp():  
 return random.randint(100000,999999)

***Step2: Sending OTP to User:***

For this step User defined function is used.

The sending otp to email address and otp sent. Message is displayed.

# For sending otp to user:  
def sending\_otp(otp,email):  
 print(f'sending\_otp {otp} to Email address')  
 print('otp sent')

***Step3: prompting the OTP:***

In this step we allow the user to enter the OTP received and allow the user to request for resending OTP. And please enter the otp sent to email address or type resend.

# For prompting the OTP and allow the user to enter the user:  
def prompt\_the\_user\_to\_enter\_otp():  
 return input(f'please enter otp sent to Email address or type resend:')

***Step4: For Verify the OTP***

In this step we verify the OTP sent and OTP entered by the user.

In this we have to observe one thing about try and except block.

# for verify the OTP:

def verify\_otp(generated\_otp,entered\_otp):  
 try:  
 entered\_otp = int(entered\_otp)  
 except ValueError:  
 return False  
 return generated\_otp == entered\_otp

***Try and Except block:***

Try and Except block uses to: The try block lets you test a block of code for errors. The except block lets you handle the error. The final block lets you execute code, regardless of the result of the try- and except blocks.

For our case checks for the user enters the correct OTP and checks for the non-numeric values.

String conversion fails than returns fail and the otp verification fails.

***Step5: Creating the main function and integrating the all above functions:***

Interconnecting and joining all functions and creating the main function.

And the no of attempts to enter the correct otp is 3.

I used to loop and condition statements.

# For prompting the OTP and allow the user to enter the user:  
# OTP verification system:  
def prompt\_the\_user\_to\_enter\_otp():  
 return input('please enter otp sent to Email address or type resend ')  
# for verify the OTP:  
def verify\_otp(generated\_otp,entered\_otp):  
 try:  
 entered\_otp = int(entered\_otp)  
 except ValueError:  
 return False  
 return generated\_otp == entered\_otp  
# Complete code with main function:  
def main ():  
 email='nonuser@gmail.com' # exmaple email  
 otp = generated\_otp()  
 sending\_otp (otp,email)  
 attempts = 3  
 while attempts > 0:  
 entered\_otp = prompt\_the\_user\_to\_enter\_otp()  
 if entered\_otp.lower() == 'resend':  
 otp = generated\_otp()  
 sending\_otp(otp, email)  
 print('A new OTP has been sent to your email address.')  
 continue  
 if verify\_otp(otp, entered\_otp):  
 print('Access Granted')  
 break  
 else:  
 attempts-= 1  
 print('Access denied')  
 print(f'Entered wrong otp please try again,You have {attempts} attempt(s) left.')  
 if attempts == 0:  
 print(f'Too many Incorrect attempts.You have {attempts} attempt(s) left.')

***Explanation of loop and condition statements:***

***Loop:***

In this I have used only while loop.

***While Loop:***

It is a control flow statement that allows to execute block of code repeatedly as a specified condition is True.

***Conditional Statement:***

In this IF conditional statement is used.

***IF:***

It will allow us to control the flow of program based on condition.

***Break and continue:***

***Break:*** Break statement stops the entire process of the loop.

***Continue***: Continue statement only stops the current iteration of the loop.

***One pre-defined function is used:***

***Pre-defined function:*** this function are given and defined by Python.

The pre-defined function used is print.

***Step6: Run the code:***

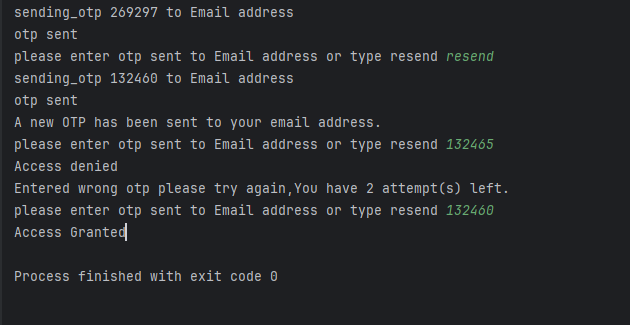
if \_\_name\_\_ == "\_\_main\_\_" is used to run the code directly.

# to run the above code:  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

Appjar is used to design the application interface.

***Output:***

The code is working perfectly.



The GUI (Graphical User Interface|) Code is provided below.

***GUI Code:***

from appJar import gui  
import random  
  
  
# OTP functions  
def generated\_otp():  
 return random.randint(100000, 999999)  
  
  
def sending\_otp(otp):  
 print(f'Sending OTP {otp} to the registered email address')  
 print('OTP sent')  
  
  
def verify\_otp(generated\_otp, entered\_otp):  
 try:  
 entered\_otp = int(entered\_otp)  
 except ValueError:  
 return False  
 return generated\_otp == entered\_otp  
  
  
# handle button events for the OTP window  
def press\_otp(button):  
 if button == "Cancel":  
 app.stop()  
 elif button == "Resend OTP":  
 resend\_otp()  
 else:  
 entered\_otp = app.getEntry("OTP")  
 if verify\_otp(app.otp, entered\_otp):  
 print('Access Granted')  
 app.infoBox("Success", "Access Granted")  
 app.stop()  
 else:  
 app.attempts -= 1  
 if app.attempts > 0:  
 app.errorBox("Error", f'Access Denied. You have {app.attempts} attempt(s) left.')  
 else:  
 app.errorBox("Error", 'Too many incorrect attempts. Access Denied.')  
 app.stop()  
  
  
# OTP prompt window  
def prompt\_otp():  
 app.otp = generated\_otp()  
 sending\_otp(app.otp)  
  
 app.setBg("White")  
 app.setFont(18)  
 app.attempts = 3  
  
 app.addLabel("otp\_title", "Enter the OTP sent to your email")  
 app.setLabelBg("otp\_title", "blue")  
 app.setLabelFg("otp\_title", "gray")  
  
 app.addLabelSecretEntry("OTP")  
  
 app.addButtons(["Submit", "Resend OTP", "Cancel"], press\_otp)  
  
 app.go()  
  
  
# Resend OTP function  
def resend\_otp():  
 app.otp = generated\_otp()  
 sending\_otp(app.otp)  
 app.infoBox("Info", "A new OTP has been sent to your email address.")  
  
  
# create a GUI variable called app  
app = gui("OTP Verification", "400x200")  
prompt\_otp()

***Future scope:***

By using SMTP Client Protocol, we can implement in online applications.