Python-Capstone Project

OTP Verification System

Nikhil Karaka
Student ID:S9807

Agenda

01 Introduction

02 Checking Test cases

03 Defining Functions

04 Conclusion

Introduction:

01

This project is about creating a OTP (One-Time Password) verification system using Python. The system generates a random 6-digit OTP and sends it to the user's email. The user then enters the OTP to verify their identity.

Technologies used:

- Google Colab
- Python
- SMTP

Importing Required Libraries

Importing Required Libraries for the OTP Verfication

```
[10] import random
     import smtplib
     from email.mime.text import MIMEText
     from email.mime.multipart import MIMEMultipart
```

This script imports necessary libraries and modules for OTP verfication:

- It uses the 'random' library for generating random values.
- It utilizes the 'smtplib' library for sending emails via the SMTP.
- The 'email.mime.text' and 'email.mime.multipart' modules are used to create email messages with text and multiple parts.

02

Defining Functions:

For this OTP Verification system project we define some functions they are:

- otp()
- send_otp_email()
- verify_otp()
- main function()

Defining Functions

```python
# Function to generate a 6-digit OTP
def otp():
    return str(random.randint(100000, 999999))

# Function to send the OTP via email
def send_otp_email(recipient_email, otp):
    sender_email = "karakanikhil2003@gmail.com"
    sender_password = "agqi vnpz oqye yxwe"
    subject = "Your OTP Verification Code"
    body = f"Your One Time Password is {otp}."

    try:
        # Setting up email
        message = MIMEMultipart()
        message['From'] = sender_email
        message['To'] = recipient_email
        message['Subject'] = subject
        message.attach(MIMEText(body, 'plain'))

        # Connecting to e-mail server
        server = smtplib.SMTP("smtp.gmail.com", 587)
        server.starttls()
        server.login(sender_email, sender_password)
        server.send_message(message)
        server.quit()

        print("OTP has been sent to your email.")
        return True
    except Exception as e:
        print(f"Failed to send OTP. Error: {e}")
        return False
```

This code defines two functions:

- otp() and send_otp_email().
- The otp() function generates a random 6-digit One-Time Password (OTP).
- The send_otp_email() function sets up the email content, connects to the Gmail SMTP server,logged in with the OTP provider sender's email and App password, and sends the email.
- This send_otp_email() function sends this OTP to a specified email address.
- If there's an error during this process, it prints the error message.

Defining Functions

```python
# Function to verify the OTP entered by the user
def verify_otp(generated_otp, max_attempts=3):
    for i in range(max_attempts):
        user_otp = input("Enter the OTP sent to your email: ").strip()
        if user_otp == generated_otp:
            print("Access Granted.")
            return True
        else:
            print("Incorrect OTP. Please try again.")
    print("Access Denied.")
    return False
```

This code defines verify_otp function:

- This code defines a function verify_otp() to verify the OTP entered by the user.
- It takes the generated OTP and allows a maximum number of attempts (default is 3).
- The user is prompted to enter the OTP, and the input is compared with the generated OTP.
- If the entered OTP matches, it prints "Access Granted." If the entered OTP is incorrect, it prompts the user to try again.
- If the maximum attempts are exceeded without a correct OTP, it prints "Access Denied."

Defining Functions

```python
if __name__ == "__main__":
    # Ask the user to enter their email address
    recipient_email = input("Enter your email address: ").strip()
    otp = generate_otp()  # Generate a 6-digit OTP

    # Try to send the OTP to the provided email
    email_sent = send_otp_email(recipient_email, otp)

    # Check if the email was sent successfully
    if email_sent:
        # If the email was sent, verify the OTP
        otp_verified = verify_otp(otp)

        if otp_verified:
            print("Verification successful! Exiting program.")
        else:
            print("Failed to verify OTP. Please try again later.")
    else:
        # If the email was not sent, ask the user to try again
        print("Failed to send OTP. Please check your email address and restart the program.")
```

This code defines main function:

- This script runs the OTP verification process when executed directly.
- It prompts the user to enter their email address, generates a 6-digit OTP, and send the OTP via email.
- If the email is sent successfully, it verifies the OTP entered by the user.
- If the OTP is verified, it prints a success message; otherwise, it prints a failure message.
- If the email fails to send, it prompts the user to check their email address and restart the program.

Checking Test cases:

03

In this OTP Verfication system project we have some test cases and scenarios to handle user entered OTP's they are:

- Scenario 1(succefull recieveing of OTP to Email and matched with OTP generated)
- Scenario 2(If entered E-mail is not Valid)
- Scenario 3(User not entering the generated password)

Checking Test cases

```
        print("Failed to verify OTP. Please try again later.")
else:
    # If the email was not sent, ask the user to try again
    print("Failed to send OTP. Please check your email address and

Enter your email address: kishorekiran1290@gmail.com
OTP has been sent to your email.
Enter the OTP sent to your email: 382784
Access Granted.
Verification successful! Exiting program.
```

Scenario 1(succefull recieveing of OTP to Email and matched with OTP generated):

- In this test case, the user entered the email address 'karakanikhil2003@gmail.com'.
- The python program generates 6-digit OTP and successfully sent it to the specified email address.
- The user then entered the OTP '350992', which matched with generated OTP.
- As a result, the function verify_otp() granted access, indicated by printing 'Access Granted.'
- The program then printed 'Verification successful! Exiting program.' and completed successfully.

Checking Test cases

```
    else:
        print("Failed to verify OTP. Please try again later.")
else:
    # If the email was not sent, ask the user to try again
    print("Failed to send OTP. Please check your email address and restart the
```

```
Enter your email address: dbjsjs
Failed to send OTP. Error: {'dbjsjs': (553, b'5.1.3 The recipient address <dbjsjs>
Failed to send OTP. Please check your email address and restart the program.
```

Scenario 2(If entered E-mail is not Valid):

- In this test case, the user entered an invalid email address 'dbjsjs'.
- The script attempted to send the OTP to the provided email address but failed.
- As a result, the function send_otp_email() printed an error message.
- The program then printed 'Failed to send OTP. Please check your email address and restart the program.'

Checking Test cases

```
# If the email was not sent, ask the user to try again
print("Failed to send OTP. Please check your email address a
```

```
Enter your email address: kishorekiran1290@gmail.com
OTP has been sent to your email.
Enter the OTP sent to your email: 443344
Incorrect OTP. Please try again.
Enter the OTP sent to your email: 224456
Incorrect OTP. Please try again.
Enter the OTP sent to your email: 115577
Incorrect OTP. Please try again.
Access Denied.
Failed to verify OTP. Please try again later.
```

Scenario 3(User not entering the generated password):

- In this test case, the user entered the email address ,and script generated a 6-digit OTP and successfully sent it to the specified email address.
- The user then entered three different OTPs, none of which matched the generated OTP.
- As a result, the function verify_otp() printed 'Incorrect OTP. Please try again.' after each incorrect attempt.
- After three failed attempts, the function verify_otp() printed 'Access Denied.' and returned False.
- The program then printed 'Failed to verify OTP. Please try again later.' indicating that the verification was unsuccessful.

04

Conclusion:

- In conclusion, this OTP (One-Time Password) verification system project efficiently generates a 6-digit OTP and sends it to the user's email address for verification.
- The system allows users to enter the received OTP and verifies it within a set number of attempts. If the OTP matches, access is granted; otherwise, access is denied.

From this project I've learnt about extra concepts like:

- OTP generation.
- E-Mail sending.
- OTP Verification.
- Error Handling etc.

Thank You