PYTHON - CAPSTONE PROJECT

OTP VERIFICATION SYSTEM

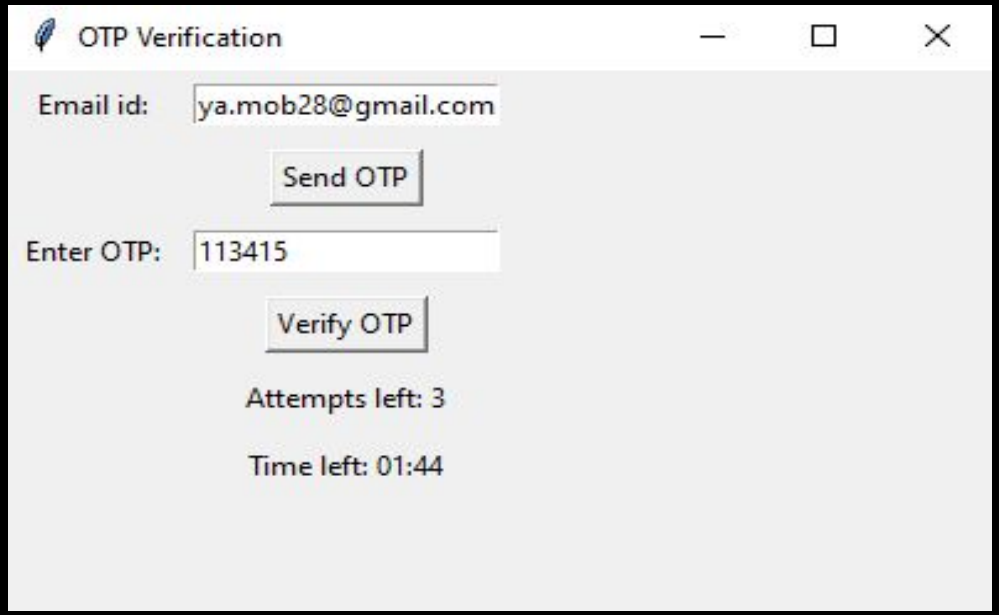
AIM:

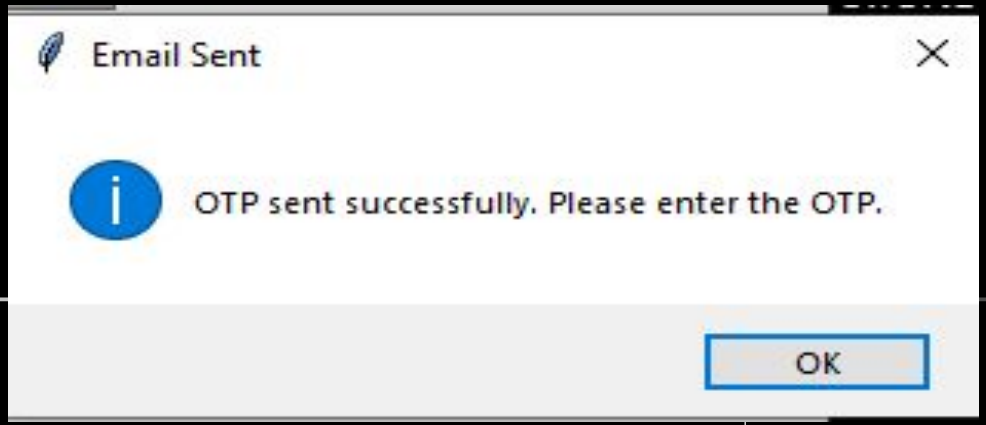
To generate a 6-digit OTP and send it to a user's email address for verification.

Upon receiving the OTP, the user should enter it into the system for validation.

If the entered OTP matches the generated OTP, access should be granted; otherwise, access should be denied.

IDE used: SPYDER





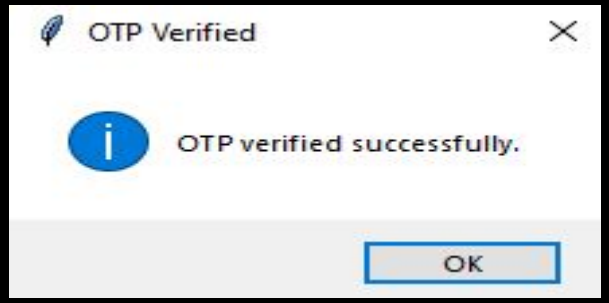
1. The GUI provides option for the user the input the email id

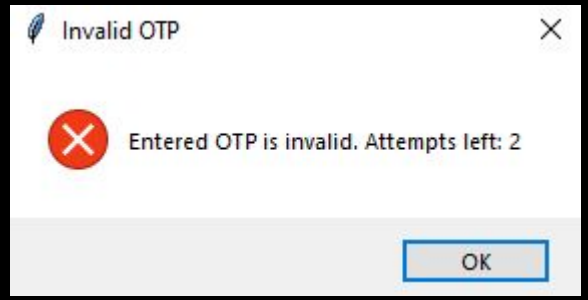
2. When the user clicks “send otp” button after entering the email id, a OTP is sent to the entered email id And a confirmation message is shown for the same

3. The user can then enter the OTP and click “Verify OTP” button

4. The user has 3 attempts, incase the user enters wrong OTP more than 3 times, they have to wait for 5 min to get a OTP again

5. The User has 3 min to enter the OTP





6. If the OTP matches the OTP sent to the mail, “OTP verified successfully” is displayed

7. If the OTP does not match the OTP sent to the mail, “Entered OTP is invaild. Attempts left” message is dispalyed

MODULES USED

```
#importing all neccessary lib and modules
import random
import smtplib
from email.message import EmailMessage
from tkinter import Tk, Label, Entry, Button, messagebox
import threading
import time
```

RANDOM: for generation of random numbers which is used for otp
SMTP: Safe Mail Transfer Protocol – Used to send emails
EmailMessage: to send a message in the email
TKINTER: to create a GUI for the user to enter the email id and OTP
THREADING: to run tasks concurrently and to create a countdown timer.
TIME: For time related functions

```python
class OTPVerificationApp:
    def __init__(self):
        self.root = Tk()
        self.root.title("OTP Verification")
        self.root.geometry("400x250")

        self.otp = None
        self.attempts = MAX_ATTEMPTS
        self.timer_running = False

        self.email_label = Label(self.root, text="Email id:")
        self.email_label.grid(row=0, column=0, padx=5, pady=5)
        self.email_entry = Entry(self.root)
        self.email_entry.grid(row=0, column=1, padx=5, pady=5)

        self.otp_label = Label(self.root, text="Enter OTP:")
        self.otp_label.grid(row=2, column=0, padx=5, pady=5)
        self.otp_entry = Entry(self.root)
        self.otp_entry.grid(row=2, column=1, padx=5, pady=5)

        self.send_otp_button = Button(self.root, text="Send OTP", command=self.send_otp)
        self.send_otp_button.grid(row=1, column=1, columnspan=2, padx=5, pady=5)

        self.verify_otp_button = Button(self.root, text="Verify OTP", command=self.verify_otp)
        self.verify_otp_button.grid(row=3, column=1, columnspan=2, padx=5, pady=5)

        self.attempts_label = Label(self.root, text=f"Attempts left: {self.attempts}")
        self.attempts_label.grid(row=4, column=1, columnspan=2, padx=5, pady=5)

        self.timer_label = Label(self.root, text="")
        self.timer_label.grid(row=5, column=1, columnspan=2, padx=5, pady=5)

        self.timer_thread = None
```

```python
def generate_otp(self):
    """Generate a 6-digit OTP."""
    return str(random.randint(000000,999999))

def send_otp(self):
    """Send OTP to the user's email."""
    self.otp = self.generate_otp()
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.starttls()
    from_mail = 'chaitanya41995@gmail.com'
    server.login('chaitanya41995@gmail.com', 'pbvr ojsv pywv ywew')
    to_mail = self.email_entry.get()

    msg = EmailMessage()
    msg['Subject'] = "OTP Verification"
    msg["From"] = from_mail
    msg["To"] = to_mail
    msg.set_content("Your OTP is: " + self.otp)

    server.send_message(msg)
    server.quit()

    self.start_timer(OTP_EXPIRY)
    self.root.after(100, lambda: messagebox.showinfo("Email Sent", "OTP sent successfully. Please enter the OTP."))
```

```
verify_otp(self):
"""Verify if entered OTP matches the generated OTP."""
entered_otp = self.otp_entry.get()
if entered_otp == self.otp:
    self.root.after(100, lambda: messagebox.showinfo("OTP Verified", "OTP verified successfully."))
    self.reset_ui()
else:
    self.attempts -= 1
    self.attempts_label.config(text=f"Attempts left: {self.attempts}")
    if self.attempts <= 0:
        self.send_otp_button.config(state="disabled")
        self.verify_otp_button.config(state="disabled")
        self.start_timer(WAIT_TIME)
        self.root.after(100, lambda: messagebox.showerror("Invalid OTP", "Maximum attempts reached. Wait for 5 minutes to try aga
    else:
        self.root.after(100, lambda: messagebox.showerror("Invalid OTP", f"Entered OTP is invalid. Attempts left: {self.attempts}
```

```python
    def start_timer(self, duration):
        """Start a countdown timer."""
        if self.timer_thread:
            self.timer_running = False
            self.timer_thread.join()  # Wait for the timer thread to finish
        self.timer_running = True
        self.timer_thread = threading.Thread(target=self.run_timer, args=(duration,))
        self.timer_thread.start()

    def run_timer(self, duration):
        """Run the countdown timer."""
        while duration > 0 and self.timer_running:
            mins, secs = divmod(duration, 60)
            time_str = "Time left: {:02d}:{:02d}".format(mins, secs)
            self.timer_label.config(text=time_str)
            time.sleep(1)
            duration -= 1
        self.timer_label.config(text="")

    def reset_ui(self):
        """Reset the UI after successful OTP verification."""
        self.otp_entry.delete(0, 'end')
        self.send_otp_button.config(state="normal")
        self.verify_otp_button.config(state="normal")
        self.attempts = MAX_ATTEMPTS
        self.attempts_label.config(text=f"Attempts left: {self.attempts}")
        self.timer_running = False

    def run(self):
        """Run the GUI."""
        self.root.mainloop()

if __name__ == "__main__":
    app = OTPVerificationApp()
    app.run()
```