‘OTP’ VERIFICATION SYSTEM

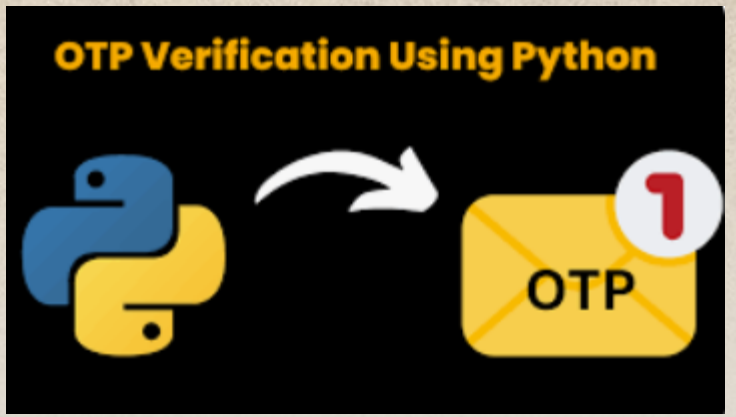
- VENNELA

OBJECTIVE: To Build An “OTP” Verification System With Python

Purpose:

- OTP verification systems serve as a crucial factor for security. It eliminates the need to remember passwords, serves as an extra layer of security, and reduces the risks of phishing.
- IDE : **Jupyter Notebook**
- Importing necessary Modules,libraries

1. random
2. SMTP (smtplib)
3. tkinter (GUI)



Requirements To Build Functionality Of Application:

- Implement a function to generate a 6-digit OTP randomly.
- Develop a function to simulate sending the OTP to the user's email address.
- Create a function to prompt the user to enter the OTP received in their email.
- Implement a function to verify if the entered OTP matches the generated OTP.
- Ensure proper error handling and user-friendly prompts throughout the system.
- Allow the user to retry OTP entry in case of incorrect input.

Explaining Each Function written In The Code

- To generate OTP , from random module **randint()** function is used.
- smtplib in Python provide server which enables connection to send emails to any valid Email-id (**Simple Mail Transfer Protocol**)
- **send_otp()** function used to send OTP to specified recipient.
- **verify-otp()** function will verify whether the user input and generated OTP are matched or not.
- resend-otp() function will send the otp again if otp does not match.

GUI (GRAPHICAL USER INTERFACE):

visual interface for a user to communicate digitally with icons, buttons, menu options.

- Python provides multiple options to develop GUI. **tkinter** is the standard application among them.
- Steps to be followed to create application :

1. Import module (from tkinter import *).
2. Create window (Tk()) which contain all GUI elements.
3. Widgets are added in between to design interface (like buttons, labels, entry fields).
4. mainloop() to run the application of the window created.

CODE

jupyter capstoneproject Last Checkpoint: 1 hour ago

File Edit View Run Kernel Settings Help

Trusted

Code JupyterLab Python 3 (ipykernel)

[239]:
```
# OTP verification
```

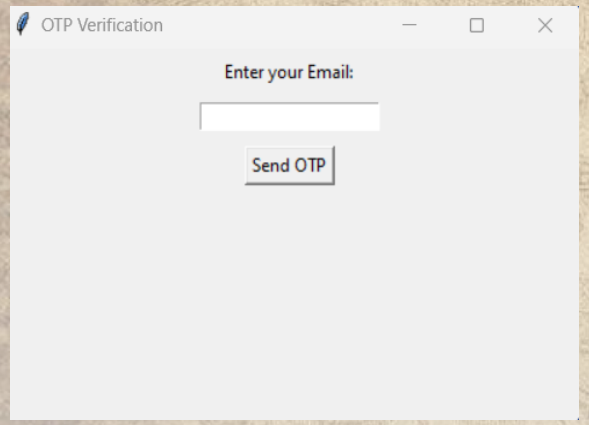
[240]:
```
from tkinter import *
```

[241]:
```
import random
import smtplib     # SMTP
import tkinter as tk  # GUI
from tkinter import messagebox    # to display message
```

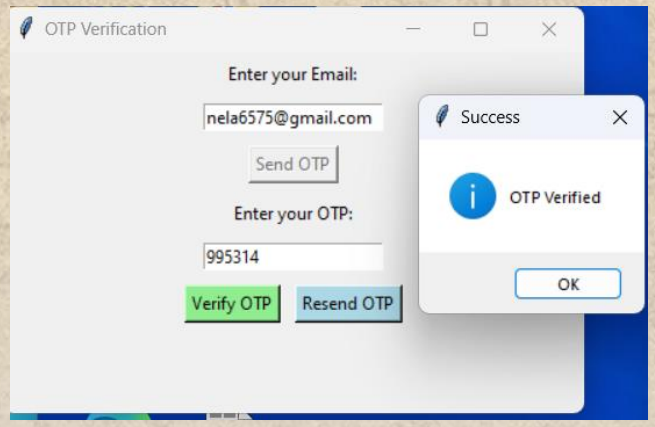
[242]:
```
# Global variable to track whether OTP has been sent
otp_sent=False
```

[243]:
```
# generate a random 6-digit OTP
def generate_otp():
    otp=""
    for i in range(6):
        otp+=str(random.randint(0,9))
    return otp
# send OTP
```

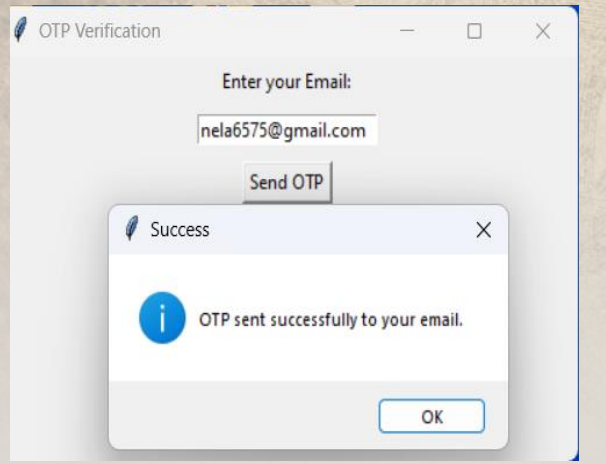
Example Output Of Verification Using OTP



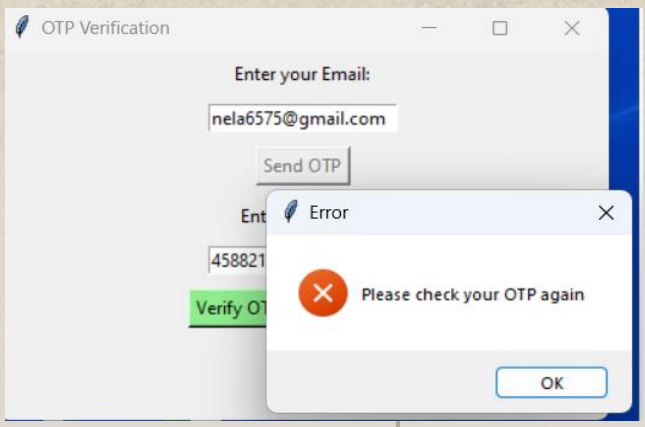
- On running the OTP verification program, displays a window to enter email-id.



- On entering the correct OTP



- Hit **Send OTP** button and check for OTP sent to mail.



- If you enter the wrong OTP, the program displays a message box saying **OTP does not match**.

RESTRICTION :

Google Colab has access to many python libraries but unfortunately it does not support GUI (Graphical User Interface) modules such as Tkinter or PyQt.
Reason is, GUI modules require a graphical display environment, which is not available in Google Colab.

TclError: no display name and no $DISPLAY environment variable

CONCLUSION:

- Developed an OTP verification system using Python and tkinter GUI for secure authentication. Users receive email for OTP. Implemented SMTP email integration for OTP delivery.

- Empowers customers to protect themselves against unauthorized transactions across Internet Banking , E-commerce , mobile banking platforms , Gmail accounts ,etc.