OTP Email Verification System using Python

1. Project Overview

This project allows a user to verify their identity using a 6■digit OTP (One■Time Password) that is generate

2. Technologies Used

- Python 3
- random – OTP generation
- smtplib – sending emails
- email.mime.text – formatting the email
- Gmail SMTP server (Port 587 with STARTTLS)

3. Features

- Random 6■digit OTP generation
- Secure email sending with starttls() encryption
- User input validation
- Three attempts to enter the correct OTP
- Custom error messages

4. Workflow

1. User enters their email address.
2. The program generates a 6■digit OTP.
3. The OTP is sent to the user's email using Gmail's SMTP server.
4. User is prompted to enter the OTP.
5. If the OTP is correct, verification is successful.
6. If the OTP is incorrect, the user has two more attempts.
7. After three failed attempts, verification is denied.

5. Code Explanation

Step 1 – Imports: random, smtplib, MIMEText.
Step 2 – generate_otp(): returns a random 6■digit string.
Step 3 – send_otp_email(): prepares the email, secures the connection with starttls(), logs in, and sends the
Step 4 – get_user_otp(): prompts until a valid 6■digit numeric OTP is entered.
Step 5 – verify_otp(): compares generated and user OTPs.
Step 6 – main(): coordinates the workflow and limits attempts to three.

6. Security Note

- Use a Gmail App Password instead of your normal Gmail password.
- starttls() encrypts the connection so credentials and email content are protected.
- OTPs are unique per session and valid for a limited number of attempts.

7. Future Enhancements

- Add OTP expiration time (e.g., 5 minutes).
- Store OTPs temporarily in a database for multi■user support.
- Provide SMS■based OTP using services like Twilio.
- Build a GUI with Tkinter or a web API with Flask/FastAPI.
- Add logging and analytics for monitoring purposes.