**Name: Khushi Manoj Ojha (S9730)**

**Python Capstone Project: OTP Verification System**

**Overview of the Project**

In a world where cybercrime is constantly evolving and fraud is becoming an everyday concern, securing sensitive information and ensuring safe online transactions has never been more critical. Every day, new methods of fraud are being devised, putting individuals and organizations at risk.

To counter these growing threats, I present to you my ‘**OTP Verification System**’ a robust and user-friendly solution designed to add an extra layer of security to your digital interactions. By generating and verifying a unique One-Time Password (OTP) for each transaction, this system helps safeguard against unauthorized access and provides a reliable defence against potential breaches.

With increasing reliance on digital platforms for everything from banking to e-commerce, this OTP system is a timely and essential tool in ensuring a safer online environment.

The OTP Verification System generates a One-Time Password (OTP), simulates sending it to the user's email, and in return verifies the OTP entered by the user. The system will provide up to 3 attempts for OTP entry.

Access is granted if the correct OTP is entered; otherwise, access is denied.

**Functions Used**

1. generate\_otp(): The main purpose is to generate a random 6-digit OTP. Using the random.randint() function to generate a number between 100000 and 999999 and ensuring that the OTP is always a 6-digit number.

In return, a 6-digit OTP as a string/result is obtained.

1. send\_otp(email, otp): The main purpose is to simulate the sending of the OTP to the user's email. It will print a message indicating the OTP is sent to the given email. In the debug mode, OTP is displayed for testing purposes.

Two parameters used are email (str) for the user's email address and otp (str) for the OTP to be sent.

1. get\_user\_otp(): The main purpose is to prompt the user to enter the OTP they received. This step uses the input () function to ask the user for the OTP.

In return, the OTP is entered by the user (as a string).

1. verify\_otp (generated\_otp, user\_otp): The main purpose is to compare the entered OTP with the generated OTP. If the OTPs match, access is granted. If they don't match, the user is prompted to try again.

The parameters used are generated\_otp (str) takes the OTP that was generated and the user\_otp (str) takes the OTP entered by the user.

Boolean value ‘True’ is returned if the OTPs match, otherwise ‘False’.

**How to run the Program**

* Steps to run the Program in Google Colab:

1. Save the Python script as otp\_verification.py on your system.
2. Visit Google Colab in your browser and sign in with your Google account if required.
3. Click on **File** → **New Notebook**.
4. Copy the OTP verification Python code and paste it into the first cell of the Colab notebook and press the **play** button on the left side of the cell.
5. Once the code is running, Google Colab will prompt you for your email address and OTP input in the output section.
6. Enter your email when prompted.
7. Enter the OTP that is displayed as the debug output or as part of the simulation.
8. After entering the OTP, Google Colab will display whether the OTP was correct or incorrect and give you the result of the verification.

* Steps to run the Program in Jupyter Notebook:

1. Install and launch Jupyter Notebook.
2. Click on New → Python 3 to create a new notebook.
3. Copy the OTP verification Python code and paste it into the first cell of the notebook and click on the **Run** button in the toolbar.
4. Jupyter will prompt you for your email address and OTP input. Since this is a simulation, you will provide your email and the OTP as shown in the debug output.
5. Once the OTP is entered, Jupyter will display the message indicating whether the OTP verification was successful or not.

**Note**: There are no external dependencies for this project, and no installation of additional libraries is required.

**Conclusion**

This OTP Verification System is simple to use, and by following the steps above, you can easily generate, send, and verify OTPs. It’s a lightweight, robust system suitable for securing access in real-world applications.