**Task 02**

**Pixel Manipulation for Image Encryption**

**Develop a simple image encryption tool using pixel manipulation. You can perform operations like swapping pixel values or applying a basic mathematical operation to each pixel. Allow users to encrypt and decrypt Images.**

### ****Introduction****

In the digital age, the security of multimedia data such as images is becoming increasingly important. One simple yet effective technique to secure image files is **pixel manipulation**. Image encryption through pixel manipulation involves modifying the pixel values of an image based on a specific algorithm and a secret key. This process transforms the original image into an unintelligible format, making it unreadable to unauthorized users.

This report presents a basic image encryption tool developed using Python, which allows users to **encrypt and decrypt images** using a user-defined key. The technique involves applying mathematical operations (such as XOR) to each pixel value in the image. Since the same key is used for both encryption and decryption, the method is classified as **symmetric encryption**.

This tool helps demonstrate the concept of image-level data security, where visual data is protected using basic cryptographic operations. While simple, it effectively shows how transformations on pixel values can help obscure and recover image data when needed.

### ****Working of the Tool:****

The image encryption tool operates via a command-line interface. Below is a demonstration of how the tool functions in practice:

#### ****Encryption Process:****

PS C:\Users\chain\OneDrive\Desktop\cyber security\image encryption and decry tool> python img.py

=== Image Encryption/Decryption Tool ===

Type 'e' to encrypt or 'd' to decrypt: e

Enter the path to the image file: you.jpg

Enter an encryption key (0-255): 123

Enter the output path for the encrypted/decrypted image: encrypted\_image1.jpg

Image successfully saved to encrypted\_image1.jpg

The user is prompted to enter:

* The operation type (e for encryption)
* The path to the input image
* A numeric encryption key (between 0 and 255)
* The output path for the encrypted image

The tool reads the image, modifies its pixel values using the key, and saves the encrypted result.

#### ****Decryption Process:****

PS C:\Users\chain\OneDrive\Desktop\cyber security\image encryption and decry tool> python img.py

=== Image Encryption/Decryption Tool ===

Type 'e' to encrypt or 'd' to decrypt: d

Enter the path to the image file: you.jpg

Enter an encryption key (0-255): 123

Enter the output path for the encrypted/decrypted image: decrypted\_image1.jpg

Image successfully saved to decrypted\_image1.jpg

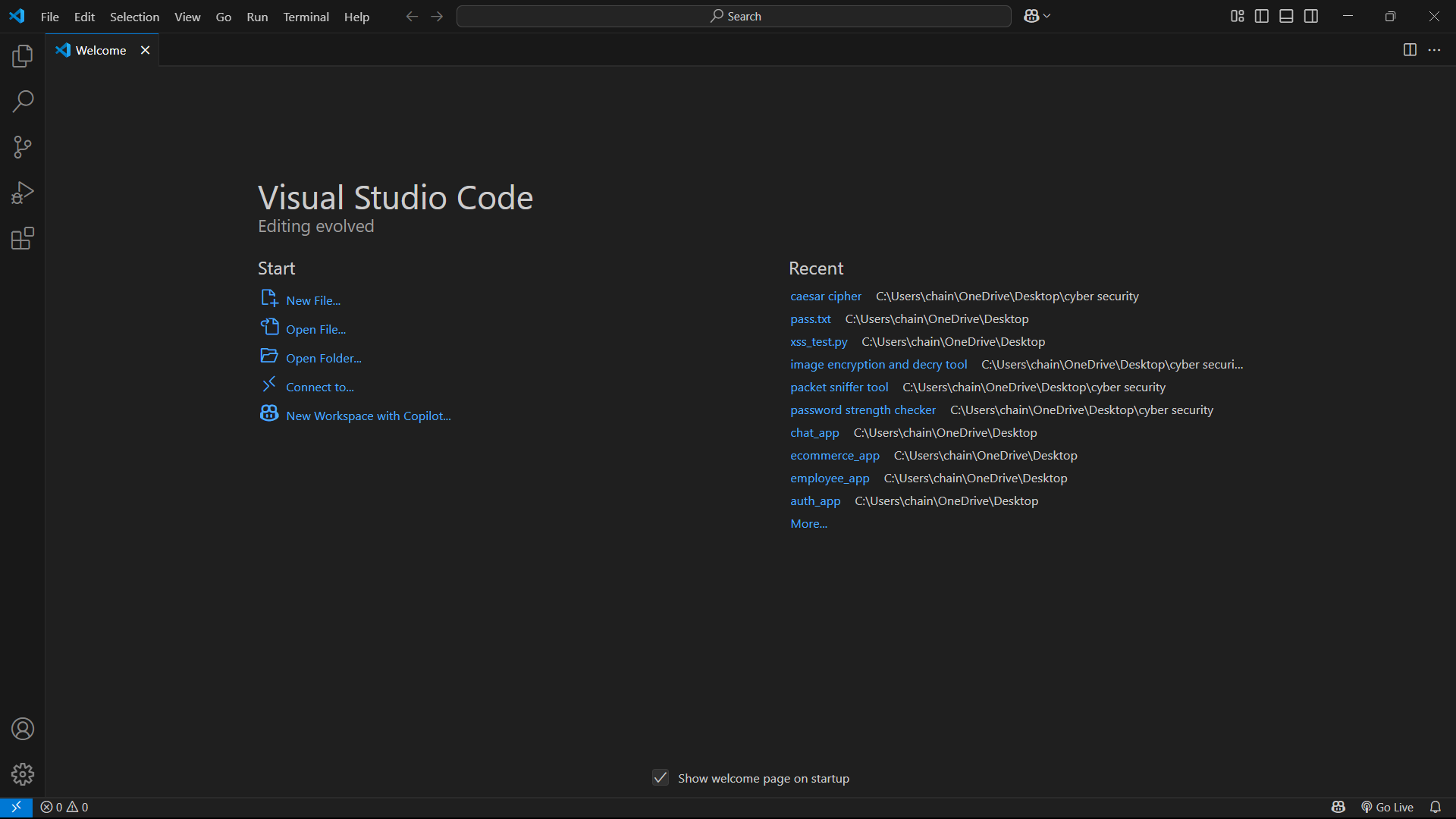
In this step, the user selects decryption (d) and provides the same key used during encryption. The tool applies the inverse operation on each pixel and reconstructs the original image, saving it to the specified output path.

**Implementation of code in Python**

**Steps: -**

1. Choose a code editor or IDE as per your choice.
2. Install and set up **VS code.**

Download: Visit <https://code.visualstudio.com/> and install VS Code.



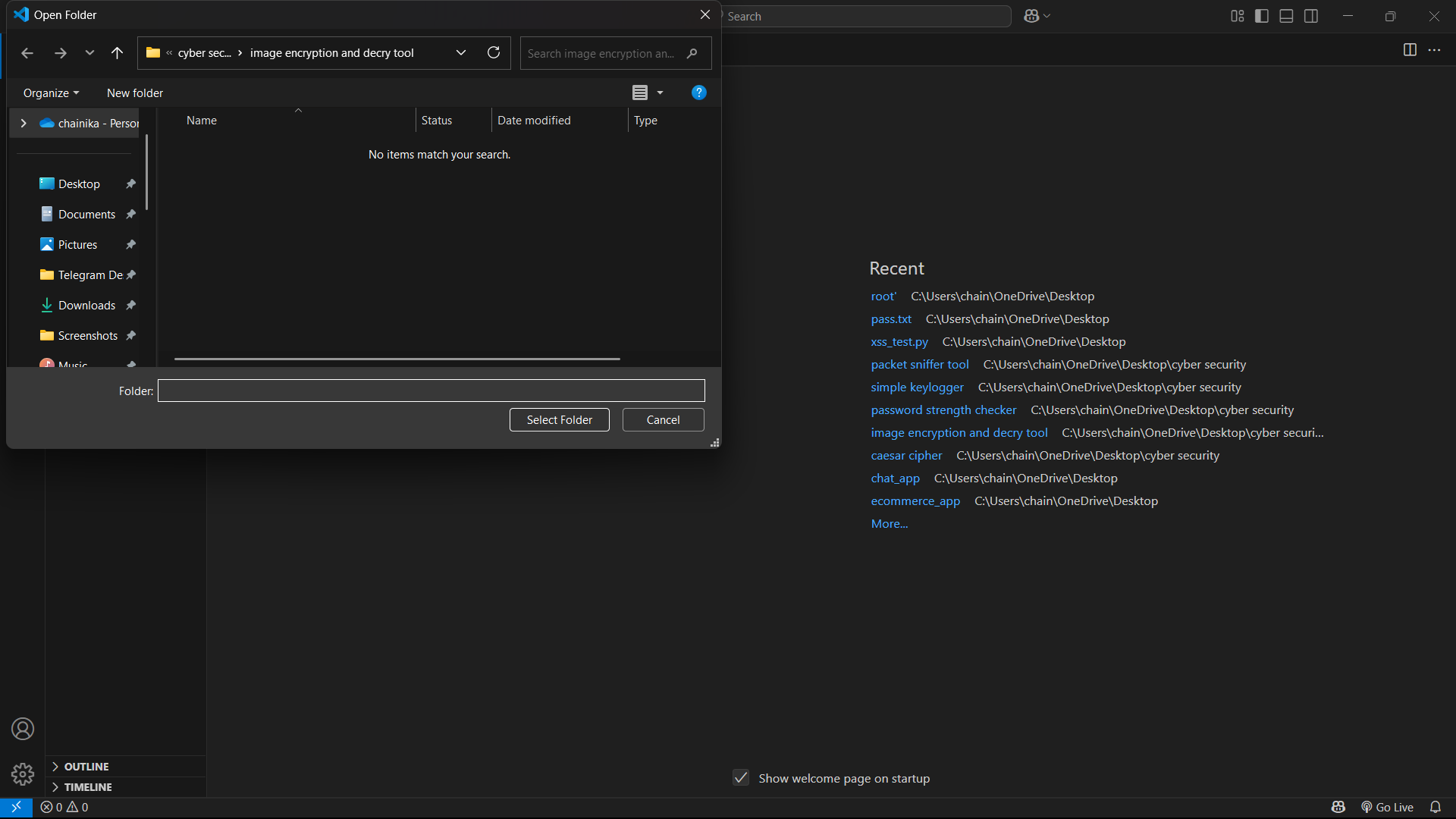
Go to extension tab (ctrl+ shift+ x) and search for Python. Click install on the official Microsoft Python extension.

Choose python interpreter (I have selected Python 3.12.7)

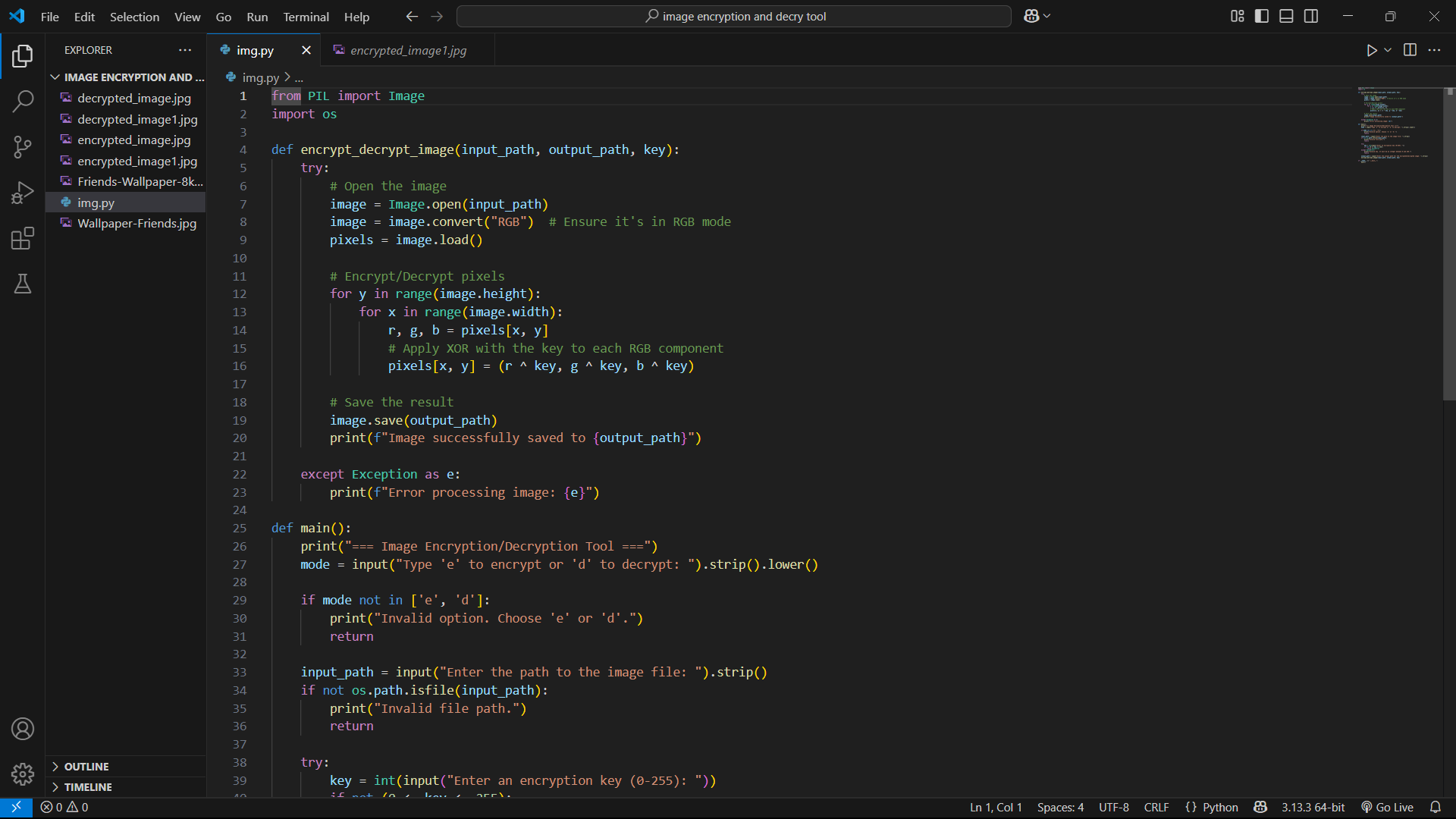
1. Create and save a python file (file having extension .py)

In VS Code: File : New File.

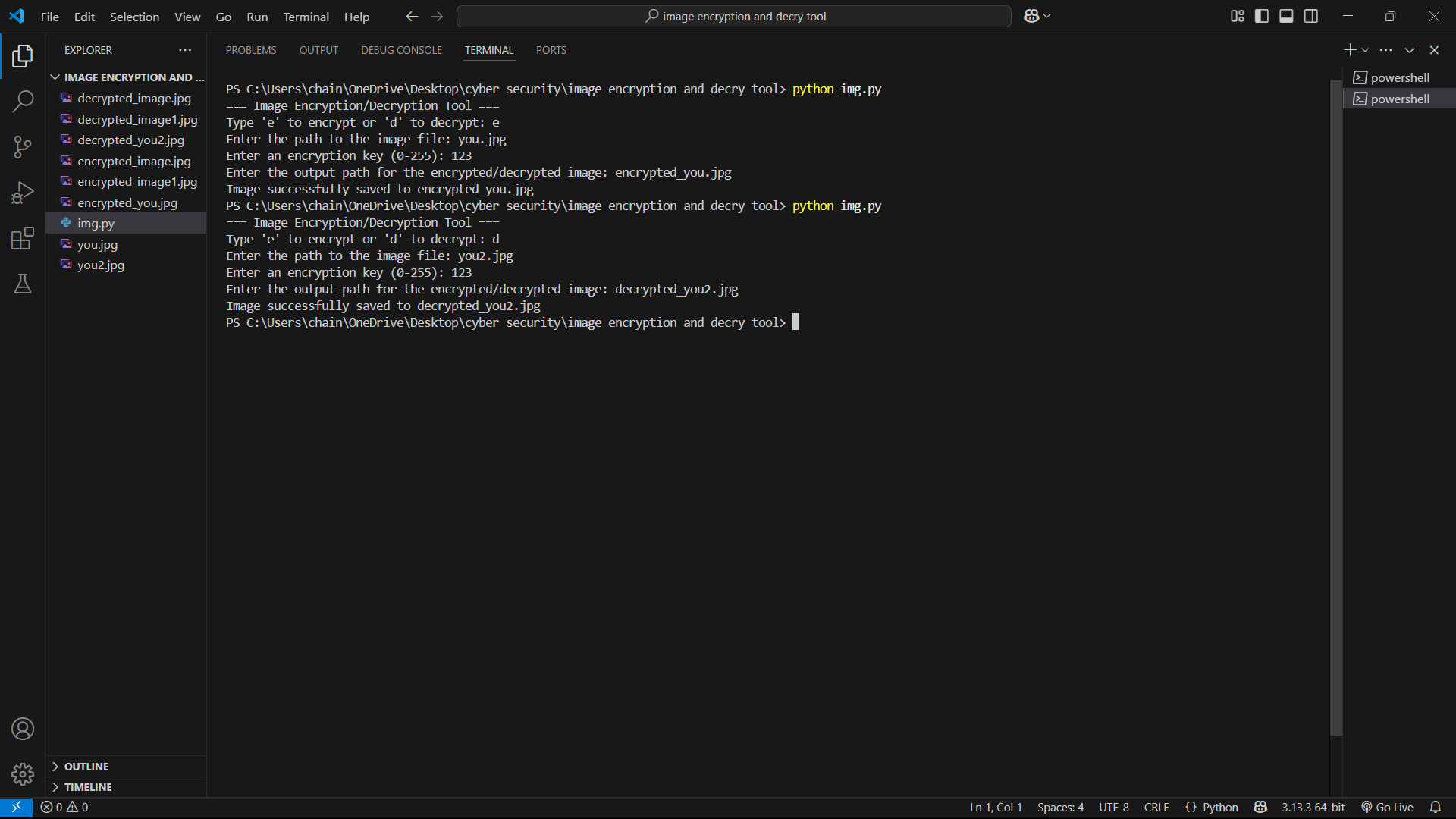
And, Save the file extension .py



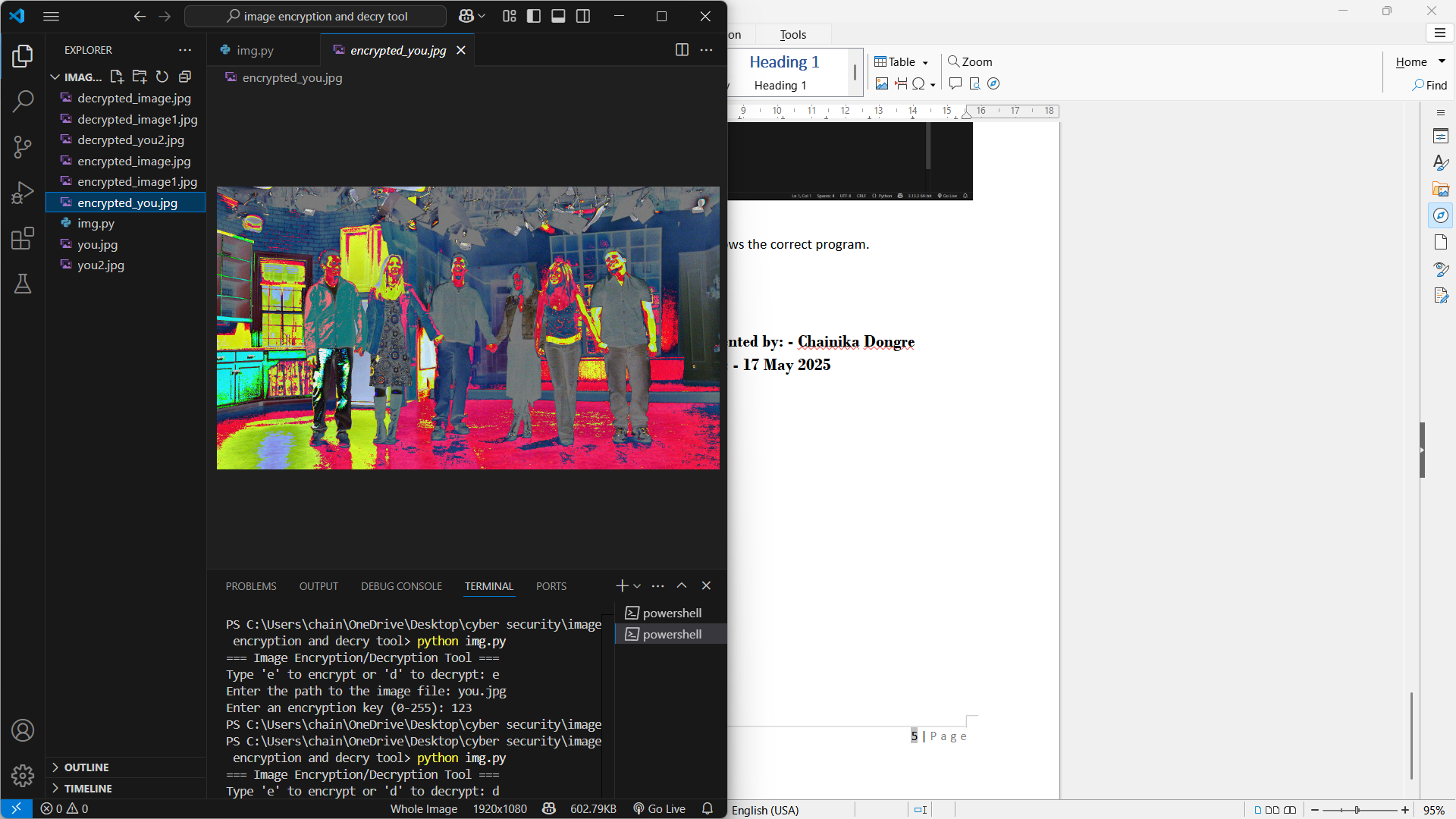
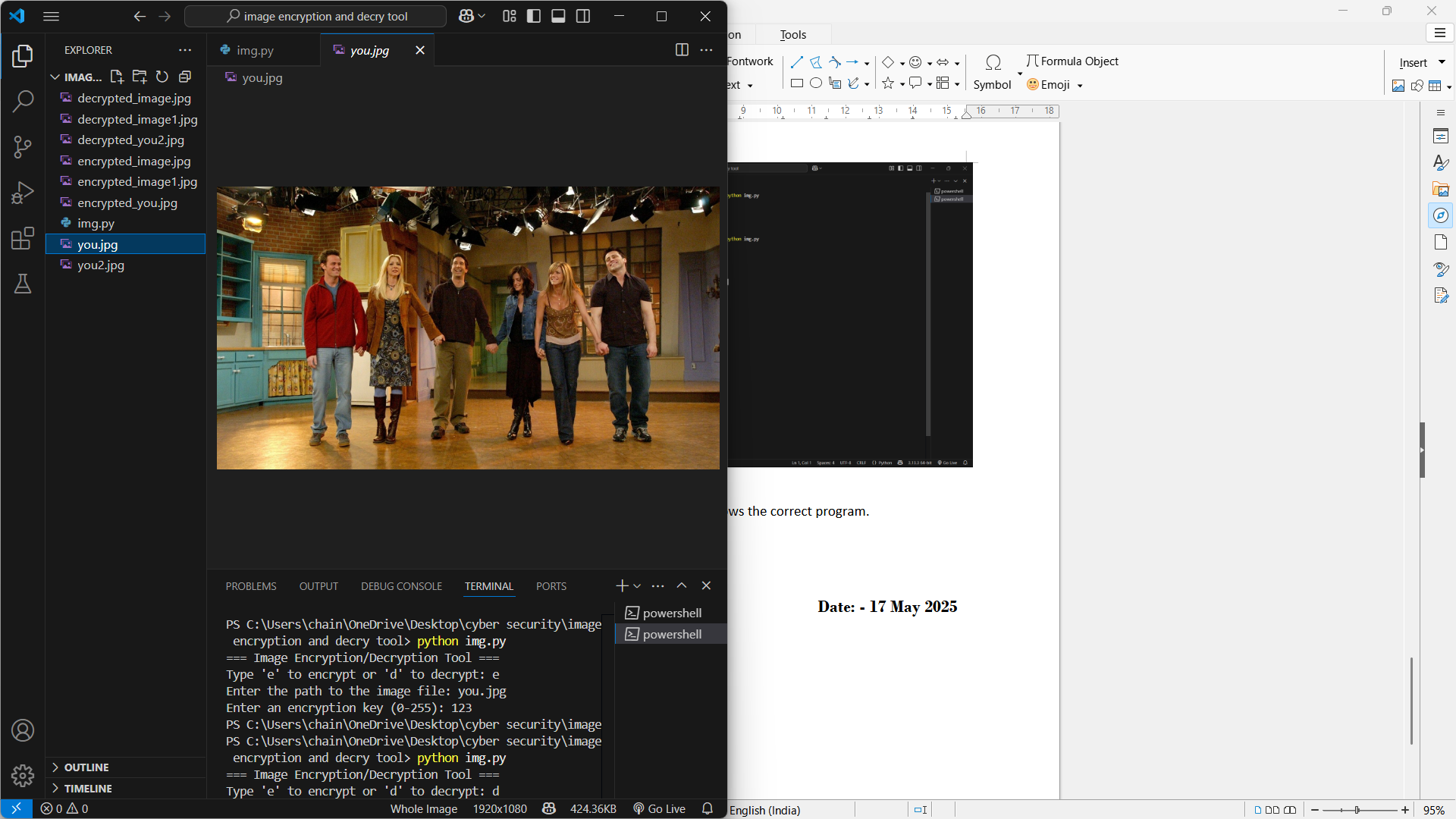
1. Initiate with writing your program:

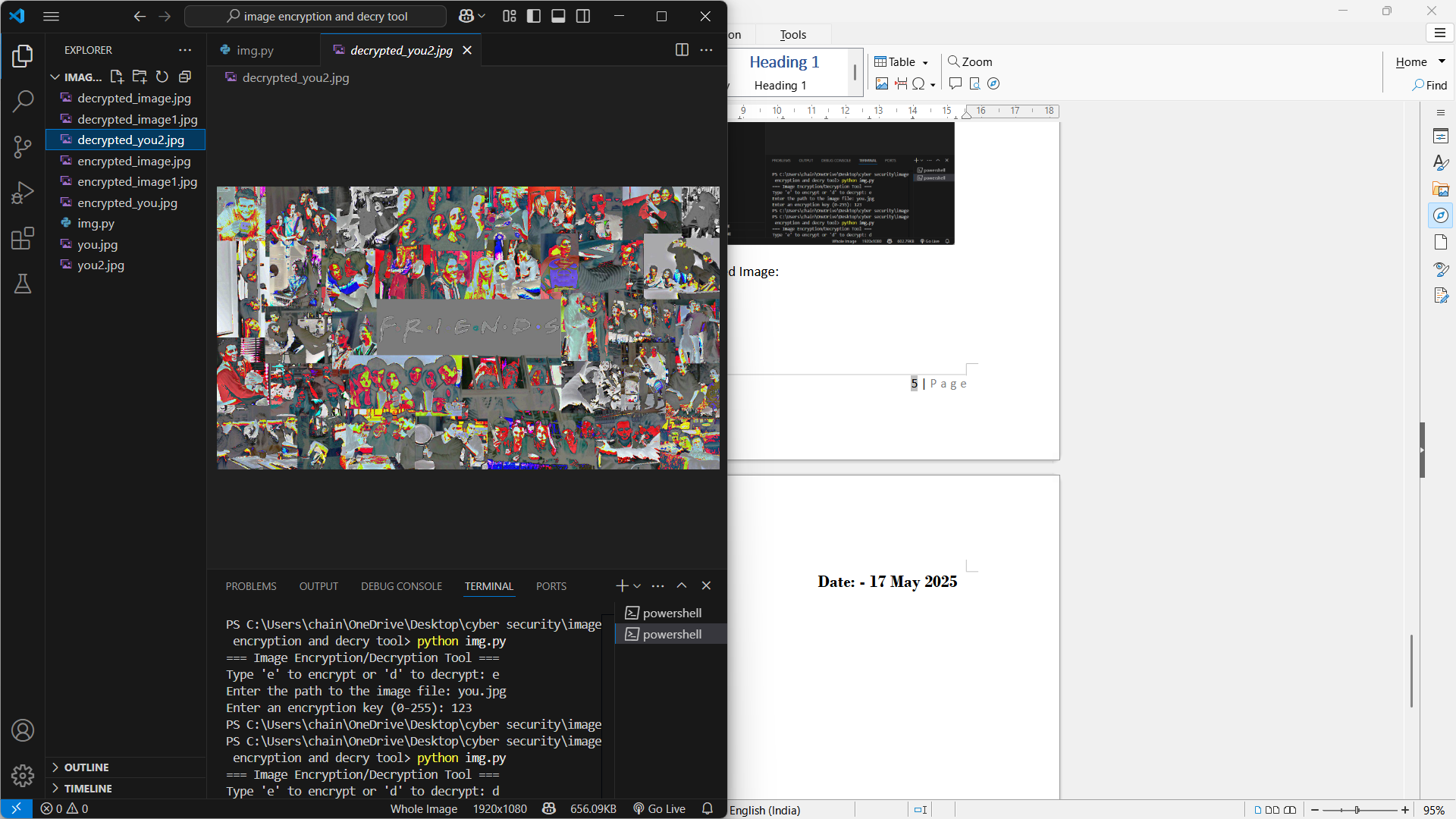
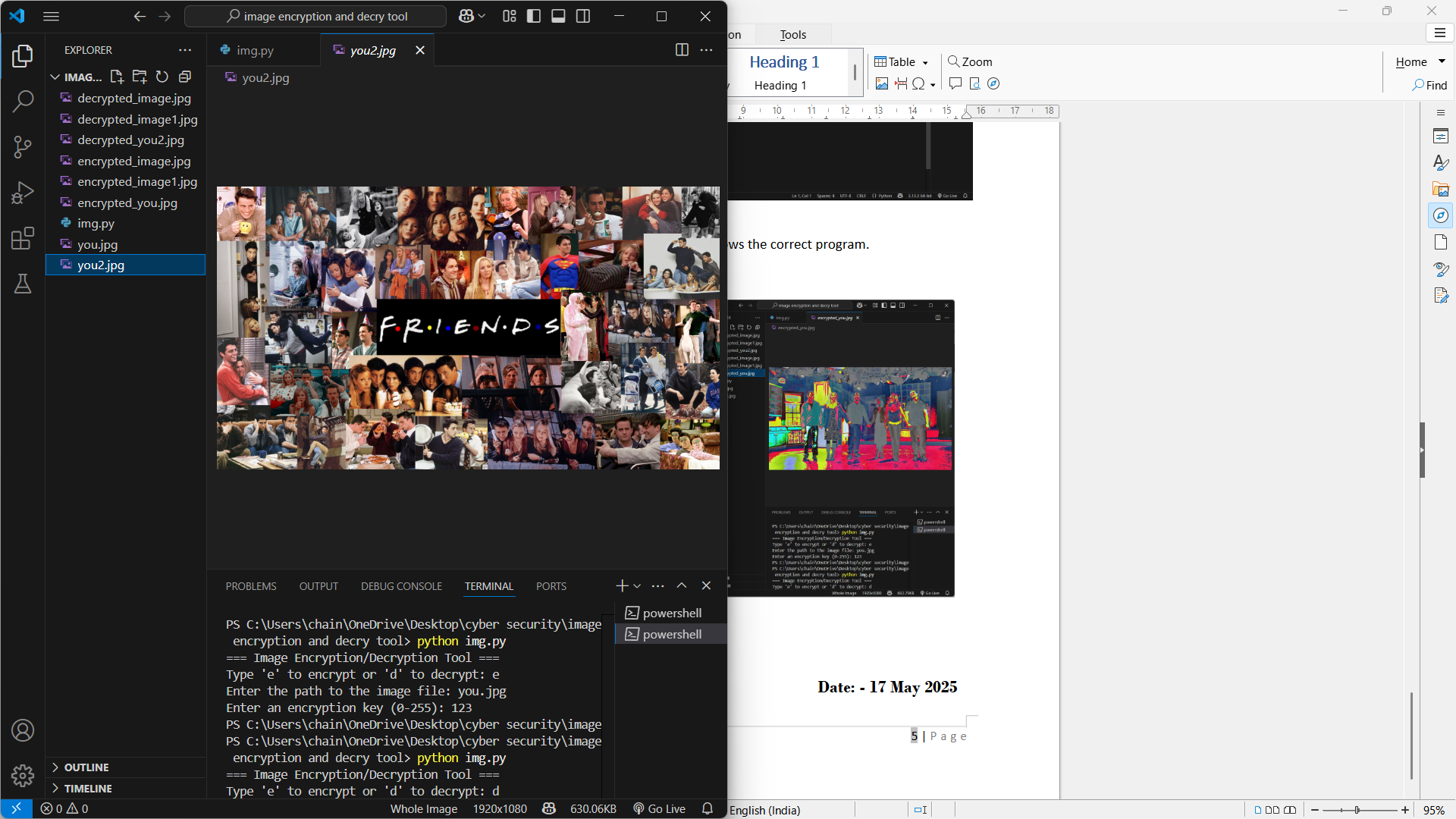


1. Then Run the program in the terminal:



**1. Encrypted Image:**

**2. Decrypted Image:**

Successful running of program with the result shows the correct program.

### ****Significance:****

The development of a basic image encryption tool using pixel manipulation holds both educational and practical importance in the field of cybersecurity. Its significance can be highlighted through the following points:

1. **Understanding Core Cryptographic Concepts**  
   This project introduces foundational concepts of encryption, such as symmetric key cryptography and data obfuscation, in a visual and interactive way. By manipulating pixel values, users gain hands-on experience with how data can be secured and later restored.
2. **Protection of Sensitive Visual Data**  
   Images often contain sensitive personal or organizational information. Encrypting images before storage or transmission helps protect against data breaches, unauthorized access, and surveillance, especially when using public or shared networks.
3. **Lightweight and Simple Security Solution**  
   The method employed is computationally lightweight, making it suitable for low-resource environments or embedded systems where complex encryption algorithms may be impractical.
4. **Foundation for Advanced Image Security**  
   Though simple, this tool lays the groundwork for more sophisticated image encryption schemes. Concepts such as key-based pixel manipulation can be expanded into more secure algorithms involving chaotic maps, permutation techniques, or hybrid cryptographic models.
5. **Real-world Applications**  
   Image encryption is applicable in many areas, including:
   * Secure sharing of medical images (e.g., X-rays, MRIs)
   * Private photo storage in cloud systems
   * Watermark protection and copyright enforcement
   * Secure communication in military or surveillance systems

**Presented by: - Chainika Dongre Date: - 17 May 2025**