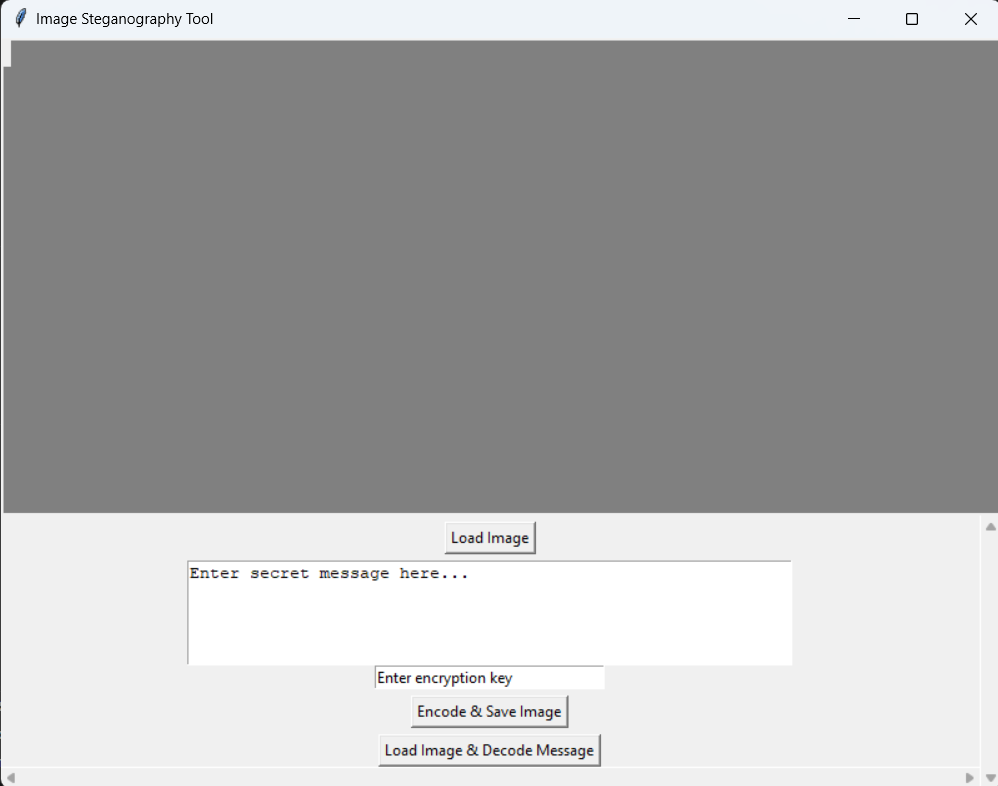
This demonstrates the complete working of our image steganography tool:

* How to **hide (encode)** messages inside images
* How to **retrieve (decode)** hidden messages using a password
* And how the tool responds to **invalid input or errors**

**1. Launch the Tool**

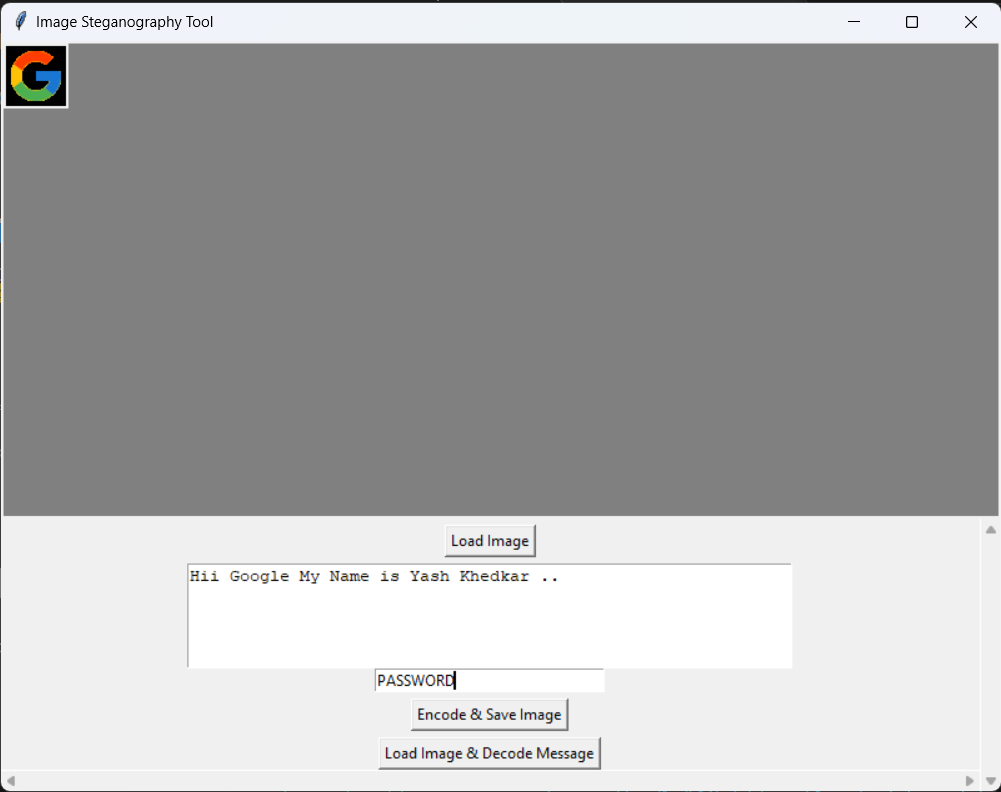
This is the UI of our **Image Steganography Tool**, designed for encoding and decoding hidden messages within images using a password-based encryption method.

### ****2. Encoding a Message into an Image****

* We selected an image (Google's logo) to hide a secret message.
* In the input text field, we entered the message:

"Hii Google My Name is Yash Khedkar .."

Then, we entered an encryption password: “PASSWORD”

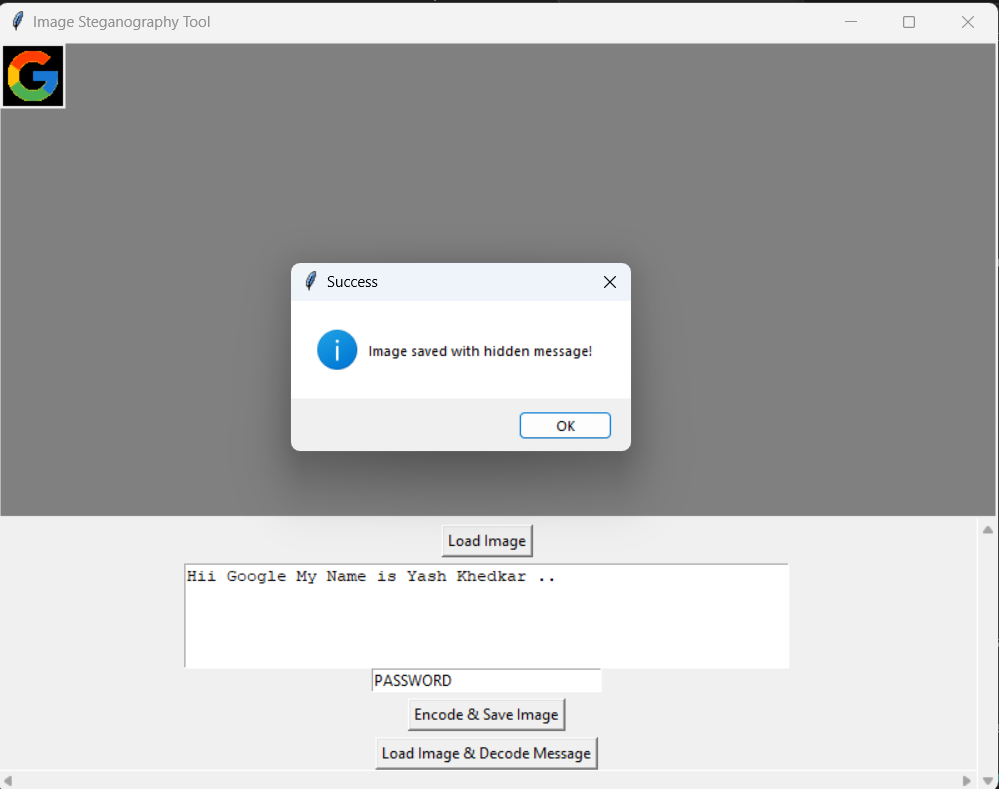


After that, we clicked on the **"Encode & Save Image"** button.

✅ A popup appeared saying: "Image saved with hidden message!"

This indicates that the image has been successfully encoded and saved.



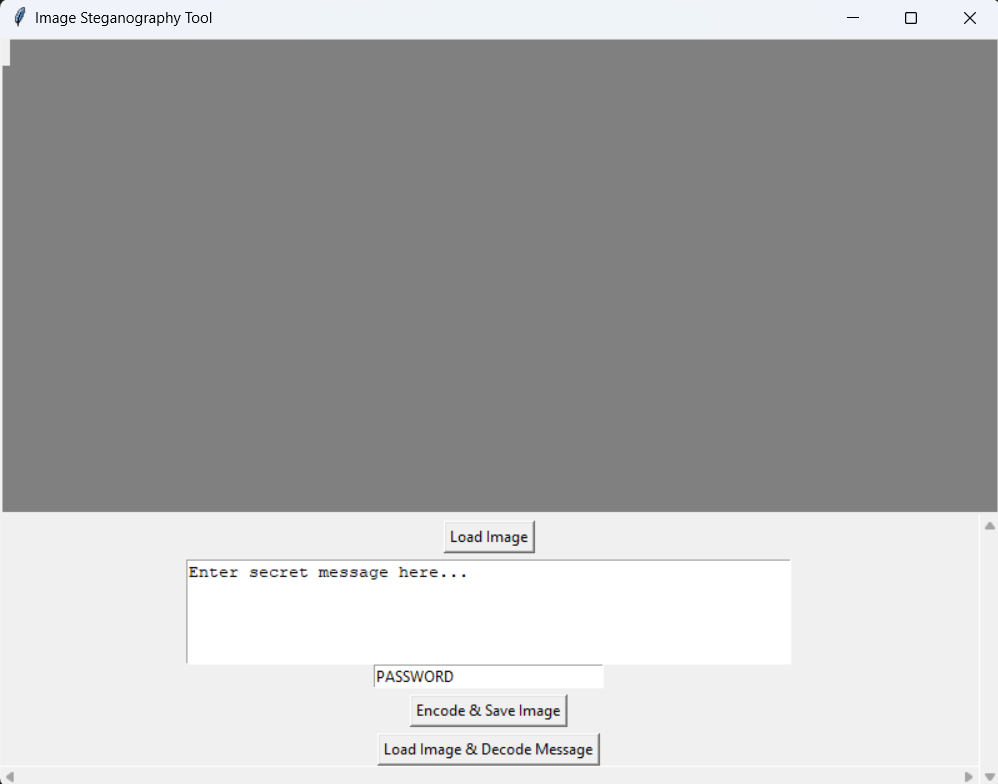


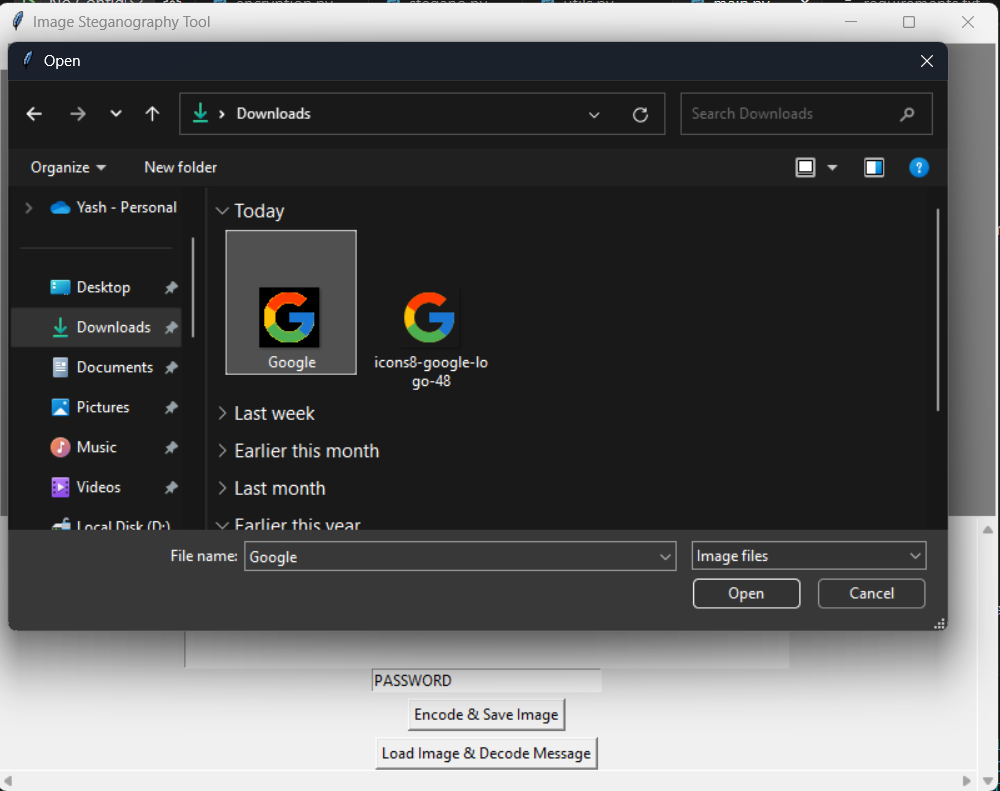
### ****3. Decoding the Message from the Image****

* We ran the tool again to decode the message.
* First, we entered the **same decryption password** used during encoding:

"PASSWORD"

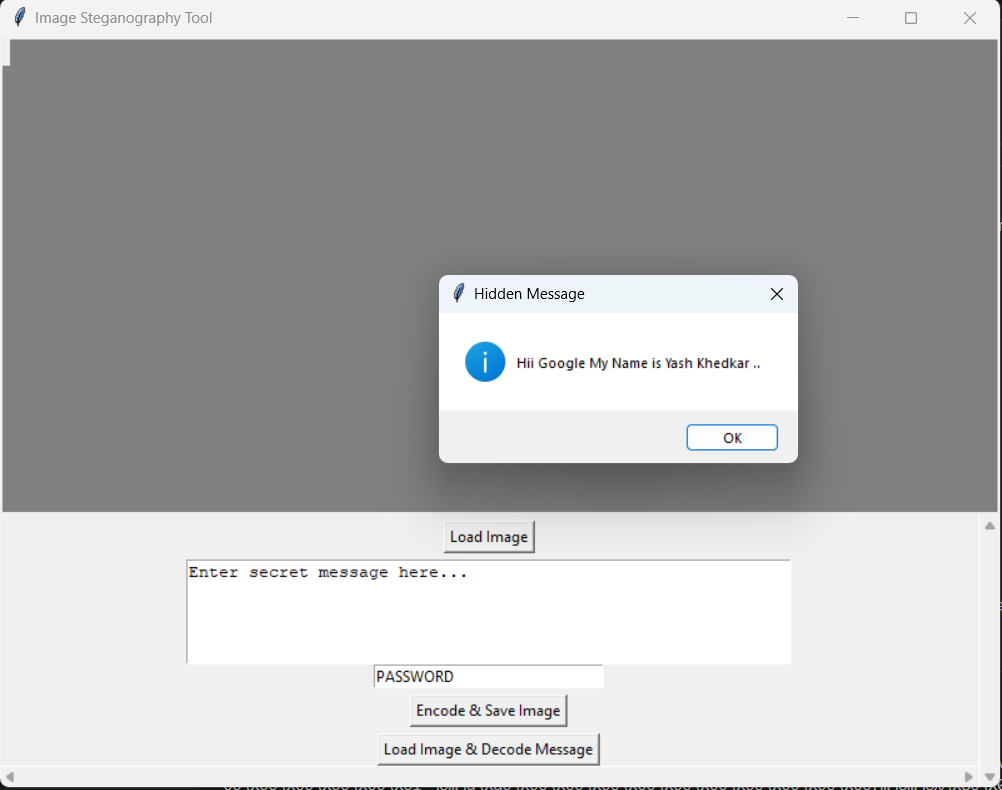
* Then we clicked on the **"Load Image & Decode Message"** button.

"

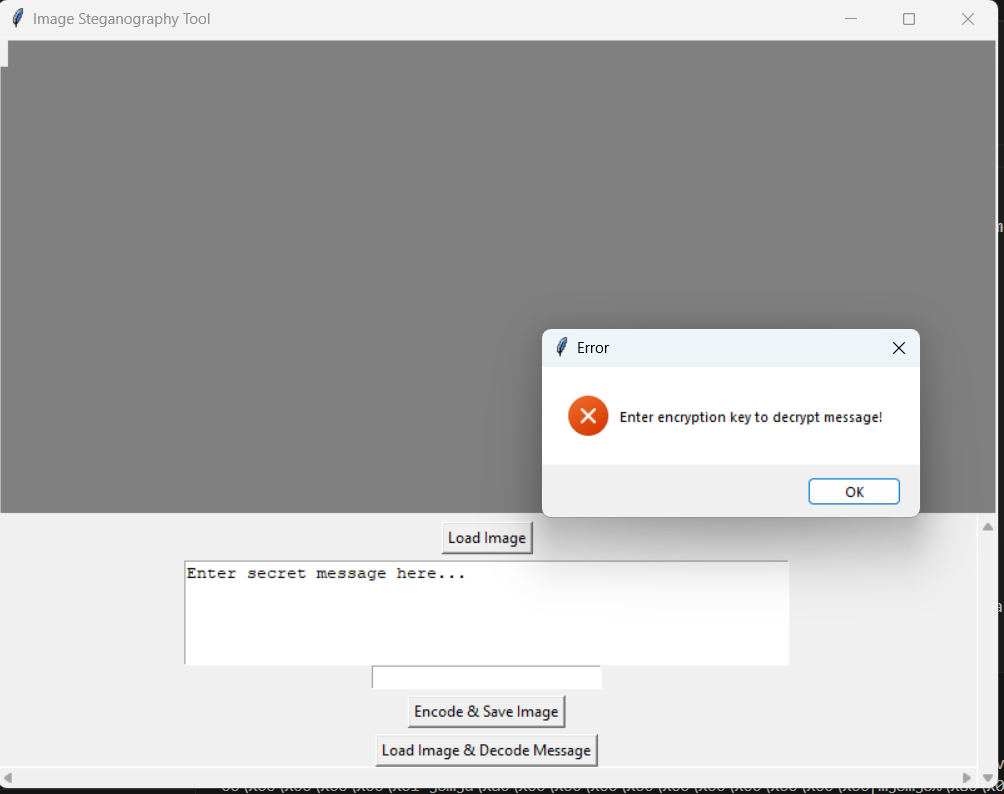
We selected the same image file saved earlier (named like google.png).

✅ The hidden message was successfully revealed:

"Hii Google My Name is Yash Khedkar ..



**4. Error Handling Scenarios**

**If the decryption key is left empty** and the user clicks on decode:  
⚠️ Popup message: "Enter encryption key to decrypt message.

**If the decryption key is incorrect**:  
❌ Popup message: "Failed to decrypt message. Possibly wrong key or corrupted data"

