

# CAPSTONE PROJECT

## PROJECT TITLE

### SECURE-DATA-HIDING-IN-IMAGE-USING- STEGANOGRAPH

**Presented By :Lokesh S. Morghade**

**Student Name : Lokesh S. Morghade**

**College Name & Department : Govindrao Wanjari College Of  
Engineering And Technology / Information Technology**

# OUTLINE

- Problem Statement
- Technology used
- Wow factor
- End users
- Result
- Conclusion
- Git-hub Link
- Future scope





---

# PROBLEM STATEMENT

In the digital era, **data security** is a critical concern. Traditional encryption techniques make data secure but are often **easily detectable**. **Steganography** is a powerful technique that allows **hiding secret messages inside images** without altering their visible appearance. The goal of this project is to implement **Least Significant Bit (LSB) steganography** with a **Graphical User Interface (GUI)** to make the encoding and decoding process **user-friendly**.

---

# TECHNOLOGY USED

- **Programming Language:** Python 
- **GUI Framework:** Tkinter 
- **Image Processing:** OpenCV 
- **Data Encoding:** Least Significant Bit (LSB) Steganography 
- **File Handling:** NumPy for efficient pixel operations

# WOW FACTORS

- ✓ **User-Friendly GUI** – Simple and interactive design for encoding and decoding messages.
- ✓ **Secure Passcode Protection** – Ensures that only authorized users can retrieve the hidden message.
- ✓ **Supports Multiple Image Formats** – Works with PNG, JPG, and JPEG images.
- ✓ **End Marker Implementation** – Prevents data corruption by marking the end of the hidden message.
- ✓ **Error Handling & Validation** – Ensures smooth execution without crashes.

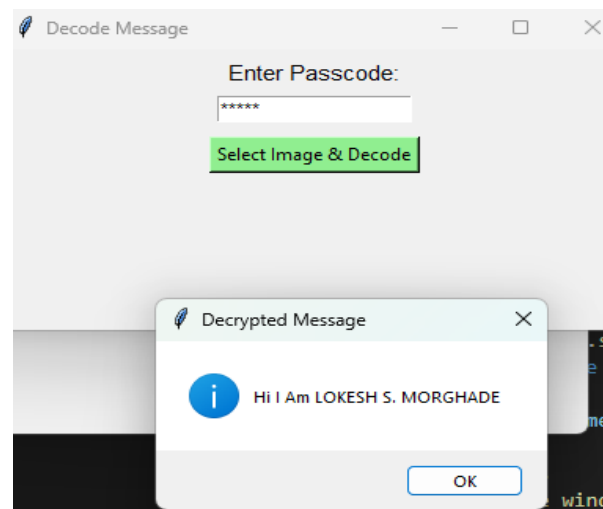
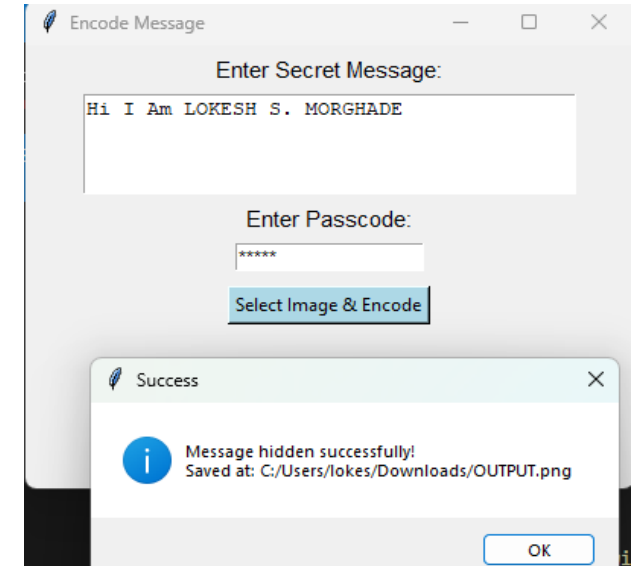
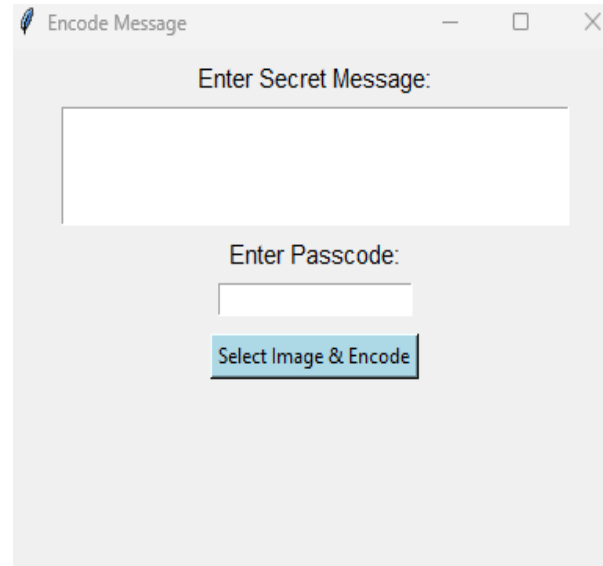
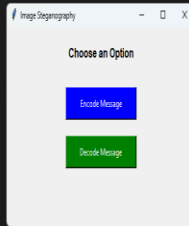
---

## END USERS

- ◆ **Journalists & Whistleblowers** – Hide sensitive information within images to prevent detection.
- ◆ **Cybersecurity Enthusiasts** – Learn and experiment with data hiding techniques.
- ◆ **Government & Intelligence Agencies** – Secure communication of confidential messages.
- ◆ **General Users** – Protect personal information by embedding messages in images.

# RESULTS

```
60: if binary_msg[:4] == "none":
61:     break
62:
63: binary_msg = binary_msg[:4]
64: decoded_msg = "".join(int.to_char(int(binary_msg[1:4]), 2) for i in range(0, len(binary_msg), 4))
65:
66: if "*" not in decoded_msg:
67:     messagebox.showerror("Error", "Corrupted data or incorrect password!")
68:     return None
69:
70: stored_password, secret_message = decoded_msg.split(":", 1)
71:
72: if stored_password != password:
73:     messagebox.showerror("Error", "Incorrect password!")
74:     return None
75:
76: return secret_message
77:
78: # All Functions
79:
80: def open_encode_window():
81:     encode_window = tk.Toplevel(root)
82:     encode_window.title("Encode Message")
83:     encode_window.geometry("480x380")
84:
85:     tk.Label(encode_window, text="Enter Secret Message:", font=("Arial", 11)).pack(pady=5)
86:     message_entry = tk.Text(encode_window, height=4, width=40)
87:     message_entry.pack()
88:
89:     tk.Label(encode_window, text="Enter Passcode:", font=("Arial", 11)).pack(pady=5)
90:     password_entry = tk.Entry(encode_window, show="*")
91:     password_entry.pack()
92:
93:     def encode_action():
94:         image_path = filedialog.askopenfilename(filetypes=(("Image files", "*.png;*.jpg;*.jpeg")))
95:
96:         if image_path:
97:             # Encode the message and save it as an image
98:             encode_message(image_path, message_entry.get(), password_entry.get())
99:
100:     tk.Button(encode_window, text="Select Image & Encode", command=encode_action).pack(pady=10)
```



---

## CONCLUSION

This project demonstrates how **steganography** can be used as a **secure method** of communication by embedding messages into images without significant distortion. The **GUI-based approach** makes it accessible to a broader audience, ensuring that even users with minimal technical knowledge can use the tool effectively.



---

## GITHUB LINK

<https://github.com/lokeshmorghade/Secure-Data-Hiding-in-Image-Using-Steganograph.git>

## FUTURE SCOPE(OPTIONAL)

- ◆ **Support for Video Steganography** – Expand from images to video frames for better data hiding.
- ◆ **Advanced Encryption Techniques** – Add AES or RSA encryption before encoding for enhanced security.
- ◆ **Mobile & Web Application** – Develop Android/iOS and web-based versions for wider accessibility.
- ◆ **Steganalysis Detection Prevention** – Implement countermeasures to evade steganalysis tools.
- ◆ **Cloud Integration** – Securely upload and process images via cloud platforms.



**THANK YOU**