**NAME- Om Sawant**

**Steganography Tools — Project Report**

---

## 🔹 1. Introduction

Steganography is the art of concealing the existence of information by embedding it within other seemingly innocuous data. This project, *Steganography Tools*, explores techniques to hide text messages inside **image**, **text**, **audio**, and **video** files, offering secure communication without arousing suspicion. The tool enables both encoding and decoding functionalities using specialized algorithms for each media type.

---

## 🔹 2. Abstract

This project implements four types of steganography:

- **Image Steganography** using Least Significant Bit (LSB) insertion.
- **Text Steganography** using Zero Width Characters (ZWCs).
- **Audio Steganography** with a modified LSB algorithm.
- **Video Steganography** combining cryptography (RC4) and steganography.

Each method converts the plaintext message into binary, then encodes it in the chosen media file (cover file), producing a stego file. The receiver can decode and retrieve the message using reverse decoding algorithms. A delimiter is added for end-of-message identification.

---

## 🔹 3. Tools Used

- **Python 3**
- **Libraries**: wave, pydub, cv2, PIL, numpy
- **Algorithms**: LSB Insertion, ZWC Encoding, Modified Audio LSB, RC4 Encryption
- **Encryption**: RC4 (for video)
- **Encoding/Decoding**: Binary manipulation, XOR logic

---

## 🔹 4. Steps Involved in Building the Project

### 🔸 A. Image Steganography

- Convert secret text to binary.
- Insert binary bits into LSBs of RGB values (Red → Green → Blue).
- Add a delimiter (111111111111) at the end.
- Decode by reading LSBs and identifying the delimiter.

🔸 **B. Text Steganography**

- Convert message using ASCII value manipulation and XOR with 10101010.
- Transform to 12-bit binary + delimiters.
- Map each 2-bit binary to a Zero Width Character (ZWC).
- Embed ZWCs after each word in cover text.

🔸 **C. Audio Steganography**

- Convert audio to 8-bit frames.
- Modify 4th and 2nd LSBs to match message bits.
- Use bitwise operations (AND, OR) for embedding.
- Decode using inverse logic and delimiters.

🔸 **D. Video Steganography**

- Encrypt the message using **RC4** stream cipher.
- Embed encrypted message into video frames using LSB.
- RC4 uses **KSA** and **PRGA** for keystream generation.
- Receiver must use the same key for decryption.

---

🔹 **5. Conclusion**

The project successfully demonstrates how steganography can be used to secure sensitive communication across multiple media formats. By combining **encryption and steganography**, especially in video, this tool ensures both secrecy and obscurity. With further enhancements, this system could support confidential operations in **military**, **intelligence**, or **forensic** contexts.

intern.mp4

Video for reference