Project report

on

Implementing Image Steganography using LSB Algorithm

A Dissertation submitted in partial fulfillment of the Academic requirements for the award of the degree of

Bachelor of Technology

In

Computer Science & Engineering

(Cyber Security)

Submitted by

B.Uday (22H51A6208)

CH.Anjali (22H51A6209)

CH.Anushka Reddy (22H51A6215)

Under the esteemed Guidance of

Dr.R.Venkateswara Reddy

(Associate Professor and HOD,CSC)

Department of Cyber Security

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(Autonomous)

(NAAC Accredited with 'A+' Grade & NBA Accredited)

(Approved by AICTE, Permanently Affiliated to JNTU Hyderabad)

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD-501401

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(Autonomous)

(NAAC Accredited with 'A+' Grade & NBA Accredited)

(Approved by AICTE, Permanently Affiliated to JNTU Hyderabad)

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD-501401

DEPARTMENT OF CYBER SECURITY

CERTIFICATE

This is to certify that the Mini Project -1 report entitled "**Implementing Image Steganography using LSB Algorithm**" being submitted by **B.Uday (22H51A6208), CH.Anjali (22H51A6209), CH.Anushka Reddy (22H51A6215)** in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering (Cyber Security)** is a record of bonafide workcarried out his/her under my guidanceand supervision.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree.

Dr. R. Venkateswara Reddy
Associate Professor & HOD
Dept. of CSC

ACKNOWLEDGEMENT

With great pleasure I want to take this opportunity to express my heartfelt gratitude to all the people `who helped in making this project a grand success.

I would like to thank **Dr. R. Venkateswara Reddy**, Head of the Department of Computer Science and Engineering, valuable suggestions and guidance during the execution of this project and for his moral support throughout the period of my study in CMRCET.

I am highly indebted to **Major Dr. V.A. NARAYANA**, Principal CMRCET, for giving permission to carry out this project in a successful and fruitful way.

I would like to thank the Teaching & Non- teaching staff of the Department of Computer Science and Engineering for their co-operation.

Finally, I express my sincere thanks to **Mr. CH. GOPAL REDDY**, Secretary, CMR Group of Institutions, for his continuous care. I sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project work.

B.Uday
(22H51A6208)

CH.Anjali
(22H51A6209)

CH.Anushka
(22H51A6215)

ABSTRACT

- Steganography is the art and science of concealing information within other innocuous data to ensure secure communication. Image steganography specifically involves embedding secret data into digital images while preserving the image's visual quality.

- This paper focuses on the implementation of a widely used technique in image steganography known as the Least Significant Bit (LSB) algorithm. Applications.

- The LSB algorithm operates by replacing the least significant bit of selected pixels in the image with secret data bits. This method is chosen for its simplicity and effectiveness in hiding information without perceptibly altering the image.

- The process involves several key steps: selecting appropriate pixels for embedding, converting secret data into binary form, and modifying the LSB of chosen pixels accordingly. Decoding the hidden message involves reversing this process, extracting the LSBs from specified pixels to reconstruct the secret data.

- This paper discusses the technical details of implementing the LSB algorithm for image steganography, including pixel selection strategies, embedding capacity considerations, and methods for ensuring robustness against common attacks such as statistical analysis and image compression.

Table Of Content

CHAPTER 1

1. INTRODUCTION

- In today's interconnected digital world, ensuring secure communication and protecting sensitive information are paramount concerns.
- Steganography, a technique dating back centuries, continues to play a crucial role in this endeavor by concealing secret data within seemingly innocuous carriers such as images, audio files, or text messages.
- The implementation of image steganography typically revolves around embedding hidden messages within digital images without perceptibly altering their visual quality.
- One of the fundamental algorithms used for this purpose is the Least Significant Bit (LSB) algorithm.
- This algorithm capitalizes on the fact that slight modifications to the least significant bit of pixel values in an image are often imperceptible to the human eye but can encode significant amounts of hidden data.

1.1 AIM

- The aim of this project is to implement the Least Significant Bit (LSB) algorithm for image steganography, focusing on embedding and extracting hidden messages within digital images while maintaining minimal perceptible distortion.
- The project aims to explore the technical intricacies of the LSB algorithm, including pixel selection strategies, encoding and decoding methodologies, and techniques to ensure the robustness and security of the hidden data against potential attacks.
- By implementing and analysing the LSB algorithm, this research aims to contribute to the understanding and practical application of steganographic techniques in securing digital communication channels effectively and efficiently.

1.2 SCOPE.

- Developing a software tool to embed and extract hidden messages using the LSB technique within digital images.
- Exploring various approaches to select pixels for embedding secret data to optimize capacity and minimize visual impact. This includes random selection, sequential selection, or strategic selection based on image characteristics.
- Designing robust methods for converting secret data into binary form and embedding it into selected image pixels using the LSB of pixel values.
- Evaluating techniques to enhance the security of embedded data against common steganalysis methods and image processing operations. Addressing challenges such as image compression, noise addition, and histogram analysis.

Image Steganography

- The word Steganography is derived from two Greek words- ‘stegos’ meaning ‘to cover’ and ‘grayfia’, meaning ‘writing’, thus translating to ‘covered writing’, or ‘hidden writing’.
- Steganography is a method of hiding secret data, by embedding it into an audio, video, image, or text file. It is one of the methods employed to protect secret or sensitive data from malicious attacks.

CHAPTER 2

2. LITERATURE REVIEW

1. Parmar Ajit Kumar Maganbhai , Prof. Krishna Chouhan-2022:

This study aims for improving the steganalysis performance and also analyzing the hiding capacities of the existing research work. The steganalysis performance of state-of-the-art detectors is near-perfect against current steganographic schemes. A novel, robust and secure hiding schemes that can resist steganalytic detection must be implemented. Hiding schemes are characterized by three complementary requirements- security against steganalysis, robustness beside distortions in the transmission channel, and capacity in terms of the embedded method.

2. Muhammad Adnan Aslam, Muhammad Rashid , Farooque Azam -2022:

An SLR has been performed to investigate the application of LSB based image Steganography. Particularly, 20 of the most relevant researches (published during 2016-2020) have been selected. Subsequently, 17 approaches/ algorithms have been recognized for image steganography. In addition, for the evaluation of image steganography techniques, 20 data sets have also been identified. Furthermore, 3 leading frameworks/ platforms/ tools for implementation of image steganography and 4 leading parameter that are frequently used to evaluate the quality of image steganography are also identified and presented.

.

CHAPTER 3

3. EXISTING SOLUTION

1. Steghide:

Steghide is a steganography program that is able to hide data in various kinds of image- and audio-files. The color- respectively sample-frequencies are not changed thus making the embedding resistant against first-order statistical tests.

Fig 1: Steghide

2. Outguess :

OutGuess is another command-line tool designed for embedding and detecting hidden data using LSB-based steganography. It focuses on preserving the statistical properties of the cover image to evade detection and includes options for passphrase protection.

Fig 2: Outguess

3.Openstego:

- **Data Hiding:** It can hide any data within a cover file .
- **Watermarking (beta):** Watermarking files with an invisible signature. It can be used to detect unauthorized file copying.

Fig 3: Openstego

4.LSB-Steganography in Programming Languages:

Many programming languages, such as Python, Java, and C/C++, have libraries and frameworks that facilitate the implementation of LSB-based steganography. These solutions often include algorithms for pixel manipulation, image file handling, and encryption/decryption functionalities.

5. Commercial Software Solutions:

Several commercial software packages offer steganographic capabilities, including LSB-based methods, as part of their suite of security and privacy tools. These solutions often integrate additional features such as batch processing, image editing tools, and compatibility with multiple image formats.

CHAPTER 4

4. PROPOSED SYSTEM

The proposed solution uses the Least Significant Bit (LSB) algorithm to embed encrypted text into an image, ensuring the text remains hidden. TripleDES encryption secures the text before embedding, using a password for protection. Users select an image, input text, and the encrypted text is stored in the image's alt attribute, making it concealed yet retrievable.

To access the hidden text, users load the image and the application extracts and decrypts the text using the same password. This ensures that only authorized individuals can view the message. The user-friendly interface allows easy image selection, text input, and downloading of modified images, combining LSB steganography and TripleDES encryption for secure information hiding

4.1 REQUIREMENT ANALYSIS

4.1.1 Software Requirements

- Web Browser
- Text Editor/IDE
- CryptoJS Library
- HTML,CSS,JavaScript

4.1.2 Hardware Requirements

- System 32 or 64 bit with 4 GB or 8 GB RAM
- CPU
- RAM

Fig 6: Web browser

Fig 7:CPU

Fig 8:RAM

4.2MERITS AND DEMERITS

Merits:

- Data Security
- Ease of Implementation
- User-Friendly Interface
- Versatility
- Non-Destructive

Demerits:

- Limited Capacity
- Browser Dependency
- Not Foolproof

CHAPTER 5

5. DESIGN DESCRIPTION

5.1 CONCEPTUAL DESIGN

The diagram shows the steps involved in Implementing image steganography using LSB Algorithm

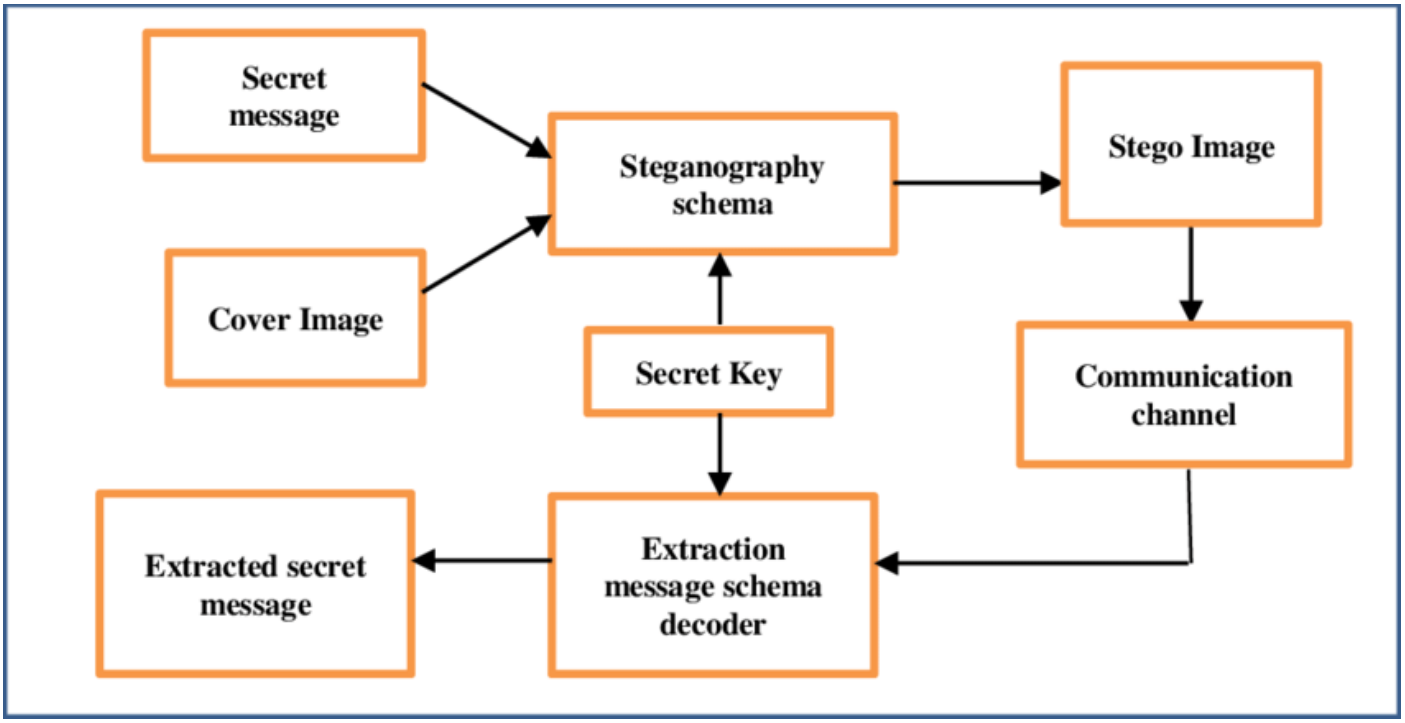

Fig 9: Conceptual design

CHAPTER 6

6.IMPLEMENTATION AND DISCUSSION

6.1 IMPLEMENTATION

Input Handling and Preparation:

The HTML provides a basic structure with inputs for selecting an image file (inputImage), entering text to encrypt (inputText), buttons to trigger encryption and decryption functions, an output img tag (outputImage) to display the image, and a download link (downloadLink) for the encrypted image

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Image Text Encryption and Decryption using TripleDES</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            text-align: center;
            padding: 50px;
            background: url('https://images.unsplash.com/photo-1510915228340-29c85a43dcfe?
crop=entropy&cs=tinysrgb&fit=max&fm=jpg&ixid=MnwzNjUyOXwwfDF8c2VhcmNofDEwfHxjeWJlcnNlY3VyaXR5fGVufDB8fHx8MTY0MzE3Mzg4OA&ixlib=rb-1.2.1&q=80&w=1080') no-repeat center center fixe
            background-size: cover;
            color: #D8BFD8; /* Light purple text color */
        }
        input[type="file"], input[type="text"] {
            display: block;
            margin: 20px auto;
            padding: 10px;
            border-radius: 5px;
            border: 1px solid #DDA0DD; /* Pale violet border */
            width: 80%;
            max-width: 400px;
            background-color: #F8F8FF; /* Ghost white background */
        }
        img {
            max-width: 300px;
            margin: 20px auto;
            display: block;
        }
        button {
            display: block;
            margin: 20px auto;
            padding: 10px 20px;
            border: none;
            border-radius: 5px;
            background-color: #800080; /* Purple button background */
            color: white;
            font-size: 16px;
            cursor: pointer;
        }
        button:hover {
            background-color: #4B0082; /* Indigo button background on hover */
        }
```

Fig 10: input handling& preparation

Encryption/Decryption:

Encryption:

Reads the selected image file using FileReader.Encrypts the entered text (inputText) using TripleDES encryption and converts it to a string.Sets the encrypted text in the alt attribute of the output image (outputImage) and displays the image with the selected file's data.Creates a download link (downloadLink) for the encrypted image.

Decryption:

Retrieves the encrypted text from the alt attribute of the output image (outputImage). Decrypts the encrypted text using TripleDES decryption and converts it back to UTF-8 format.Displays an alert with the decrypted text.

```
        a:hover {
            background-color: #4B0082; /* Indigo link background on hover */
        }
    </style>
</head>
<body>
    <h1>Image Text Encryption and Decryption using LSB Algorithm</h1>
    <input type="file" id="inputImage" accept="image/*">
    <input type="text" id="inputText" placeholder="Enter text to encrypt">
    <button onclick="encryptAndEmbedText()">Encrypt and Embed Text</button>
    <button onclick="retrieveAndDecryptText()">Retrieve and Decrypt Text</button>
    <br>
    <img id="outputImage" src="" alt="Output Image">
    <a id="downloadLink" href="#" download="encrypted_image.png" style="display:none;">Download Encrypted Image</a>

    <!-- Including CryptoJS library for encryption and decryption -->
    <script src="https://cdnjs.cloudflare.com/ajax/libs/crypto-js/4.1.1/crypto-js.min.js"></script>
    <script>
        const password = 'your_password'; // Define your encryption password

        function encryptAndEmbedText() {
            const fileInput = document.getElementById('inputImage');
            const textInput = document.getElementById('inputText').value;
            const file = fileInput.files[0];

            if (!file || !textInput) {
                alert('Please select an image file and enter text.');
                return;
            }

            const reader = new FileReader();
            reader.onload = function(event) {
                const imageData = event.target.result;

                // Encrypt the text using TripleDES
                const encryptedText = CryptoJS.TripleDES.encrypt(textInput, password).toString();

                // Display the image and set the encrypted text in the alt attribute
                const outputImage = document.getElementById('outputImage');
                outputImage.src = imageData;
                outputImage.alt = encryptedText;

                // Create a downloadable link for the image
                const downloadLink = document.getElementById('downloadLink');
                downloadLink.href = imageData;
```

Fig 11:encryption/decryption

Using password for authentication:

To set up a password input field for encryption, where users can enter a password before encrypting the text and embedding it into the image, you can modify the HTML and JavaScript as follows

```
// Store encryption password globally
encryptionPassword = encryptPasswordInput;

const reader = new FileReader();
reader.onload = function(event) {
    const imageData = event.target.result;

    // Encrypt the text using TripleDES
    const encryptedText = CryptoJS.TripleDES.encrypt(textInput, encryptionPassword).toString();

    // Display the image and set the encrypted text in the alt attribute
    const outputImage = document.getElementById('outputImage');
    outputImage.src = imageData;
    outputImage.alt = encryptedText;

    // Create a downloadable link for the image
    const downloadLink = document.getElementById('downloadLink');
    downloadLink.href = imageData;
    downloadLink.style.display = 'block';
};
reader.readAsDataURL(file);
```

Fig 12:Password for authentication

CHAPTER 7

7. RESULT

we have successfully Implemented text encryption and decryption within image metadata using LSB algorithm.

Website interface:

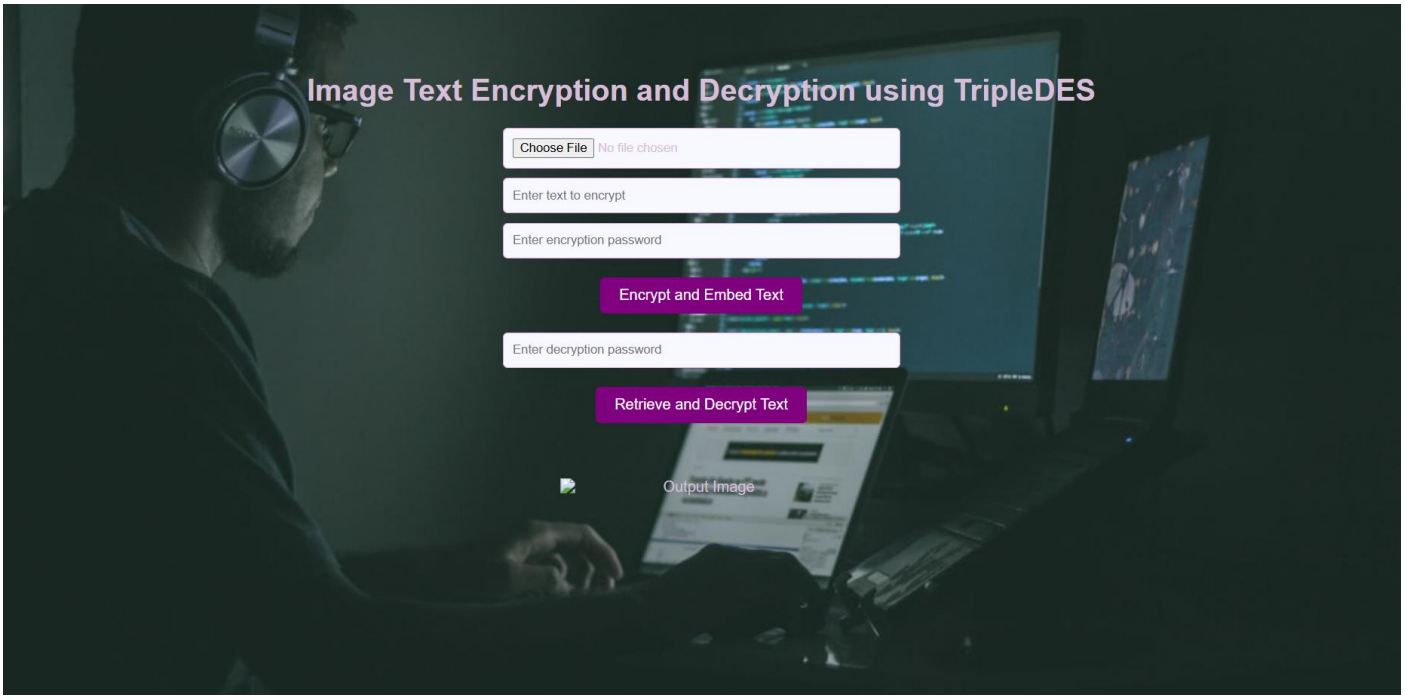

Fig 13

Uploading the image and Encrypting message:

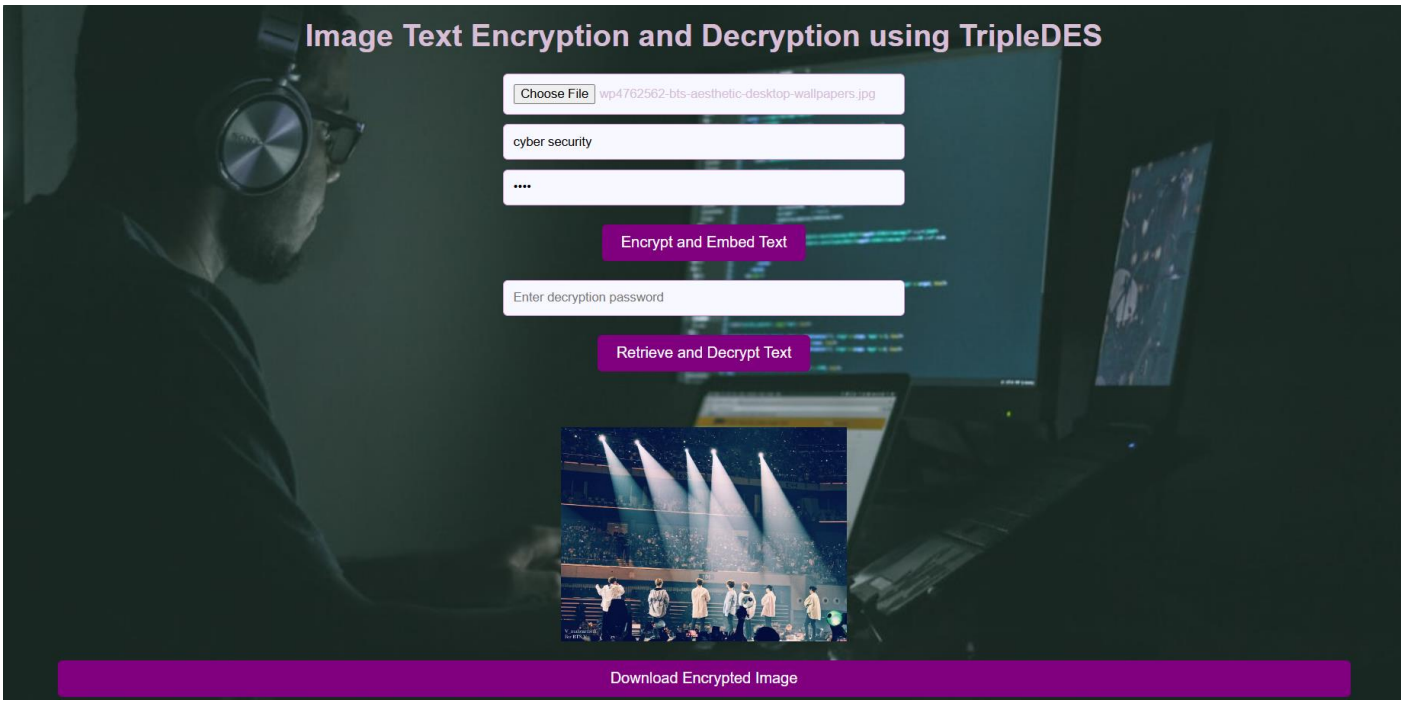

Fig 14

Uploading the image and Decrypting message:

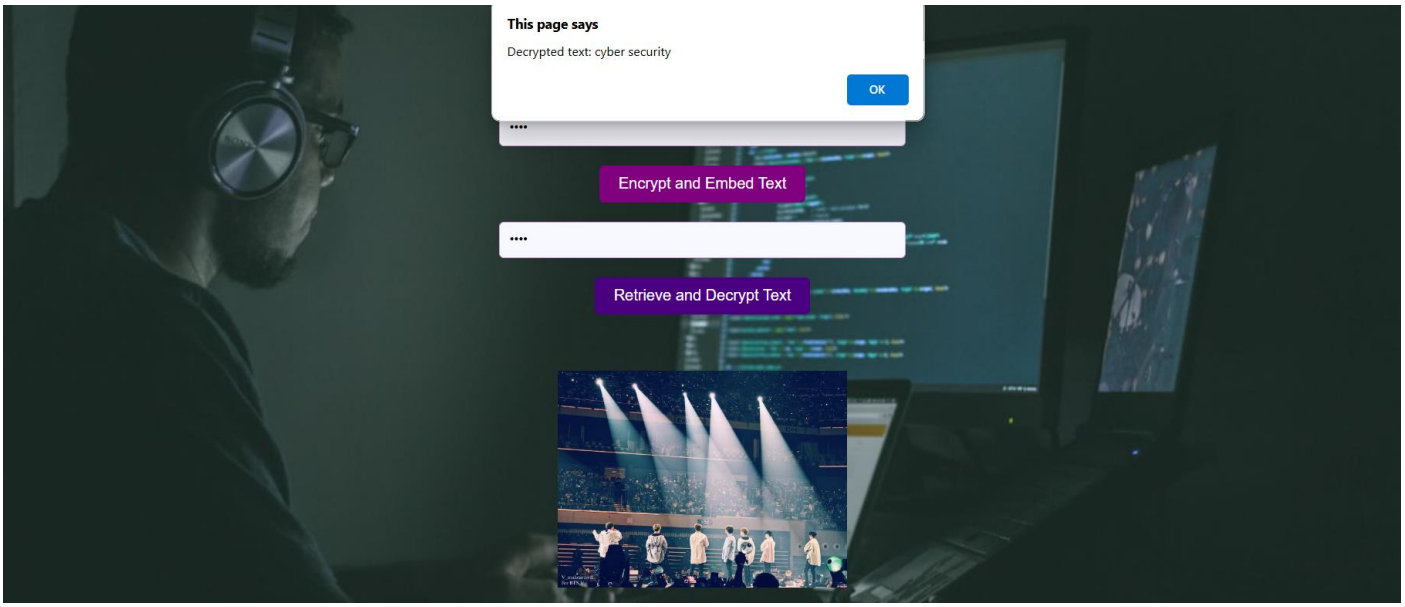

Fig 15

CHAPTER 8

8.CONCLUSION AND FUTURE ENHANCEMEMT

8.1. CONCLUSION

- The project "Image Steganography using LSB Algorithm" demonstrates a practical application of steganography techniques in digital image processing. By successfully embedding and extracting hidden messages within images using the Least Significant Bit (LSB) method, we have shown that it is possible to transmit information securely and discreetly.
- Throughout the project, we explored the fundamentals of image processing, the principles of steganography, and the specific implementation of the LSB algorithm.
- The system we developed ensures that the hidden data is imperceptible to the human eye, maintaining the visual integrity of the image while providing a covert channel for communication.
- Overall, this project serves as a foundational step in understanding and applying steganography, opening up opportunities for further research and development in the field of secure information transmission.

8.2 FUTURE ENHANCEMENTS

➢ Increased Robustness:

- **Error Correction Codes:** Integrate error correction codes (like Hamming codes or Reed-Solomon codes) to ensure the integrity of hidden data, even if the image undergoes some modifications or distortions.
- **Enhanced LSB Techniques:** Explore advanced LSB techniques such as LSB matching, which can provide better resistance against statistical attacks.

➢ Security Enhancements:

- **Encryption:** Combine LSB steganography with encryption techniques to add an extra layer of security, ensuring that the hidden data is not only concealed but also encrypted.
- **Steganalysis Resistance:** Implement methods to make the embedded data resistant to steganalysis attacks, which are designed to detect the presence of hidden data.

8.3 REFERENCES

1. Johnson, N. F., & Jajodia, S. (1998). "Exploring Steganography: Seeing the Unseen." IEEE Computer, 31(2), 26-34.

2. Fridrich, J. (2009). "Steganography in Digital Media: Principles, Algorithms, and Applications." Cambridge University Press.

3. Katzenbeisser,S., & Petitcolas, F. A. P. (2000)."Information Hiding Techniques for Steganography and Digital Watermarking." Artech House.

4. Provos, N., & Honeyman, P. (2003)."Hide and Seek: An Introduction to Steganography." IEEE Security & Privacy, 1(3), 32-44.

5. Wayner,P.(2002)."Disappearing Cryptography: Information Hiding: Steganography & Watermarking." Morgan Kaufmann Publishers.