

DATA ENCODING USING VISUAL CRYPTOGRAPHY

Enrollment numbers	21103042	21103150	21103160
Names of students	Astha Raghuwanshi	Jahanvi Gupta	Ragini Mittal
Name of supervisor	Ms. Preeti Mittal		



May - 2024

Submitted in the partial fulfillment of the Degree of

Bachelor of Technology

in

Computer Science Engineering

**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING &
INFORMATION TECHNOLOGY**

JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY , NOIDA

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Date	2024-05-09		
Place	Jaypee Institute of Information Technology, Sector 62, Noida		
Name of supervisor	Ms. Preeti Mittal		
Signature of supervisor			
Enrollment numbers	21103042	21103150	21103160
Names of students	Astha Raghuwanshi	Jahanvi Gupta	Ragini Mittal
Signature of Students			

CERTIFICATE

This is to certify that the work titled **Data Encoding using Visual Cryptography** submitted by our group in partial fulfillment for the **B.Tech** Computer Science of Jaypee Institute of Information Technology, Noida has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Signature of supervisor	
Name of supervisor	Ms. Preeti Mittal
Designation	Assistant Professor (Grade-II)
Date	2024-05-09

ACKNOWLEDGEMENT

We would like to express our sincere thanks and gratitude to our supervisor **Ms. Preeti Mittal** for her willingness to give us valuable advice and direction whenever we approached her with a problem. We are also grateful to our mentor **Nishita Gulati** for her constant mentorship and support throughout the course of the project.

Your valuable guidance and support helped us in various phases of completion of this project. We will always be thankful to you in this regard.

Enrollment numbers	21103042	21103150	21103160
Names of students	Astha Raghuwanshi	Jahanvi Gupta	Ragini Mittal
Signature of Students			
Date	2024-05-09		

SUMMARY

Our project is a web application developed using Streamlit in Python, aiming to enhance information security through a combination of LSB (Least Significant Bit) steganography and visual cryptography techniques. In the encoding phase, the application takes an input image and text to be concealed. LSB steganography involves subtly adjusting the least significant bits of colored pixels in the image, embedding the text message within the image without perceptible alterations.

Following the steganographic process, visual cryptography is employed. The encoded image is divided into 'n' number of shares using visual cryptography. Each share independently reveals no information about the original image or message, but when a minimum number of 'k' ($k \leq n$) shares are superimposed, the visual cryptography mechanism reconstructs the original image, making it visually perceptible. This process ensures that even (k-1) shares disclose any sensitive information.

For decoding, the web application prompts the user to upload the required number of shares of the visual cryptography scheme. Upon overlap, the shares reconstruct the hidden image. Internally, LSB steganography is applied again to extract the concealed text message from the reconstructed image. This dual-layered security approach combines the obscurity of LSB steganography with the cryptographic resilience of visual cryptography, providing a robust means of safeguarding secrets. The innovative fusion of these techniques fortifies data protection by making it challenging for unauthorized entities to discern both the existence of concealed information and its content.

Name of supervisor	Ms. Preeti Mittal		
Signature of supervisor			
Enrollment numbers	21103042	21103150	21103160
Names of students	Astha Raghuwanshi	Jahanvi Gupta	Ragini Mittal
Signature of Students			
Date	2023-05-09		

TABLE OF CONTENTS

<u>Chapter No.</u>	<u>Topics</u>	<u>Page No.</u>
Chapter-1	Introduction	8 - 23
	1.1 General Introduction	
	1.2 Problem Statement	
	1.3 Significance of the problem	
	1.4 Brief description of solution approach	
	1.5 Comparison of existing approaches to the problem framed	
Chapter-2	Literature Survey	24- 27
	2.1 Literature review	
	2.2 Integrated summary of the literature studied	
Chapter-3	Requirement Analysis and Solution Approach	28 - 32
	3.1 Overall description of the project	
	3.2 Requirement Analysis	
	3.3 Solution Approach	
Chapter-4	Modeling and Implementation details	33 - 42
	4.1 Design diagrams	
	4.1.1 Use Case diagram	
	4.1.2 Control Flow Diagrams	
	4.1.3 Class Diagram	
	4.1.4 Activity Diagram	
	4.2 Implementation details and issues	
	4.3 Risk Analysis and Mitigation	

Chapter-5	Testing	43 - 50
	5.1 Testing Plan	
	5.2 Component decomposition and type of testing required	
	5.3 List of test cases	
	5.4 Error and Exception Handling	
	5.5 Limitations of the solution	
Chapter- 6	Findings, Conclusion, and Future Work	51 - 52
	6.1 Findings	
	6.2 Conclusion	
	6.3 Future Work	
	References	53

CHAPTER-1

INTRODUCTION

1.1 GENERAL INTRODUCTION

In our increasingly interconnected digital landscape, the need for cryptographic techniques has become paramount. Cryptography serves as the safeguard for sensitive information in the face of evolving cybersecurity threats. As data transmission and storage have become integral aspects of modern communication, the vulnerability of information to unauthorized access and manipulation has grown exponentially. Cryptographic techniques provide a robust defense mechanism by transforming data into an unintelligible format, ensuring that even if intercepted, the content remains confidential. This heightened security extends beyond simple confidentiality; cryptography also verifies the integrity of data, confirming that it has not been altered during transmission. Moreover, cryptographic protocols facilitate secure authentication, allowing parties to establish trust and verify identities in digital interactions. Whether applied to secure financial transactions, protect personal information, or fortify communication channels, cryptographic techniques form an essential toolkit for upholding privacy, integrity, and authenticity in the dynamic and interconnected digital landscape.

Digital Image Processing

Digital Image Processing means processing digital image by means of a digital computer. We can also say that it is a use of computer algorithms, in order to get enhanced image to extract some useful information.

Image processing mainly include the following steps:

1. Importing the image via image acquisition tools;
2. Analyzing and manipulating the image;
3. Output in which the result can be altered image or a report which is based on analyzing that image.

What is an Image?

An image is defined as a two-dimensional function, $F(x, y)$, where x and y are spatial coordinates, and the amplitude of 'F' at any pair of coordinates (x, y) is called the intensity of that image at that point. When x , y , and amplitude values of 'F' are finite, we call it a digital image.

In other words, an image can be defined by a two-dimensional array specifically arranged in rows and columns.

Digital Image is composed of a finite number of elements, each of which elements have a particular value at a particular location. These elements are referred to as picture elements, image elements, and pixels. A Pixel is most widely used to denote the elements of a Digital Image.

Representation of an Image in a Matrix

As we know, images are represented in rows and columns we have the following syntax in which images are represented:

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & f(1,2) & \dots & f(1,N-1) \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & f(M-1,2) & \dots & f(M-1,N-1) \end{bmatrix}$$

The right side of this equation is digital image by definition. Every element of this matrix is called image element, picture element, or pixel.

Types of Images

1. BINARY IMAGE– The binary image as its name suggests, contains only two pixel elements i.e. 0 & 1, where 0 refers to black and 1 refers to white. This image is also known as Monochrome.
2. BLACK AND WHITE IMAGE– The image which consists of only black and white color is called black and white image.
3. 8 bit COLOR FORMAT– It is the most famous image format. It has 256 different shades of colors in it and is commonly known as Grayscale Image. In this format, 0 stands for Black, and 255 stands for white, and 127 stands for grey.
4. 16 bit COLOR FORMAT– It is a color image format. It has 65,536 different colors in it. It is also known as High Colour Format. In this format the distribution of color is not as same as Grayscale image. A 16 bit format is actually divided into three further formats which are Red, Green and Blue which is the famous RGB format.

What is Visual Cryptography?

Visual cryptography is a cryptographic technique which allows visual information (pictures, text, etc.) to be encrypted in such a way that the decrypted information appears as a visual image. One of the best-known techniques has been credited to Moni Naor and Adi Shamir, who developed it in 1994[7]. They demonstrated a visual secret sharing scheme, where an image was broken up into n shares so that only someone with all n shares could decrypt the image, while any $n - 1$ shares revealed no information about the original image. Each share was printed on a separate transparency, and decryption was performed by overlaying the shares. When all n shares were overlaid, the original image would appear.

In this example, the image has been split into two component images. Each component image has a *pair* of pixels for every pixel in the original image. These pixel pairs are shaded black or white according to the following rule: if the original image pixel was black, the pixel pairs in the component images must be complementary; randomly shade one ■□, and the other □■. When these complementary pairs are overlapped, they will appear dark gray. On the other hand, if the original image pixel was white, the pixel pairs in the component images must match: both ■■ or both □□. When these matching pairs are overlapped, they will appear light gray.

So, when the two component images are superimposed, the original image appears. However, without the other component, a component image reveals no information about the original image; it is indistinguishable from a random pattern of ■□ / □■ pairs. Moreover, if you have one component image, you can use the shading rules above to produce a *counterfeit* component image that combines with it to produce any image at all.

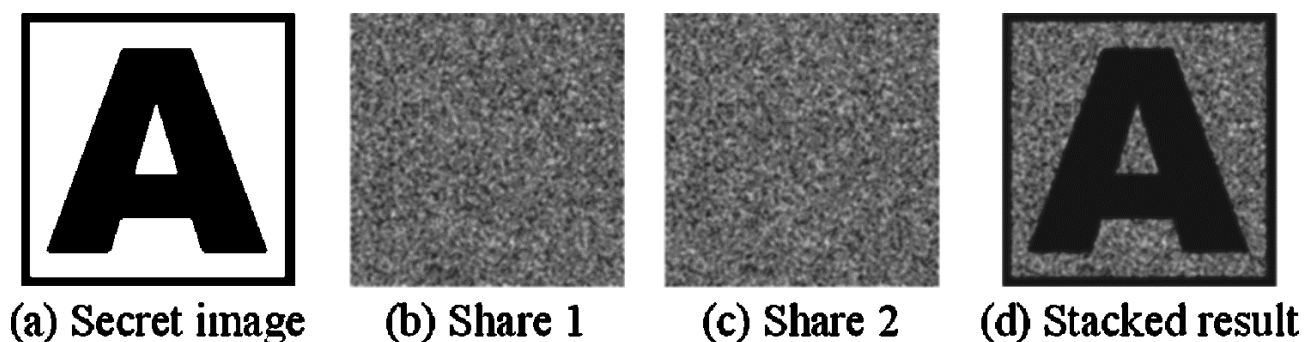


Figure 1.1: Visual Cryptography example

Binary Image

A binary image is a digital image that has only two possible values for each pixel. Typically, the two colours used for a binary image are black and white. The colour used for the object(s) in the image

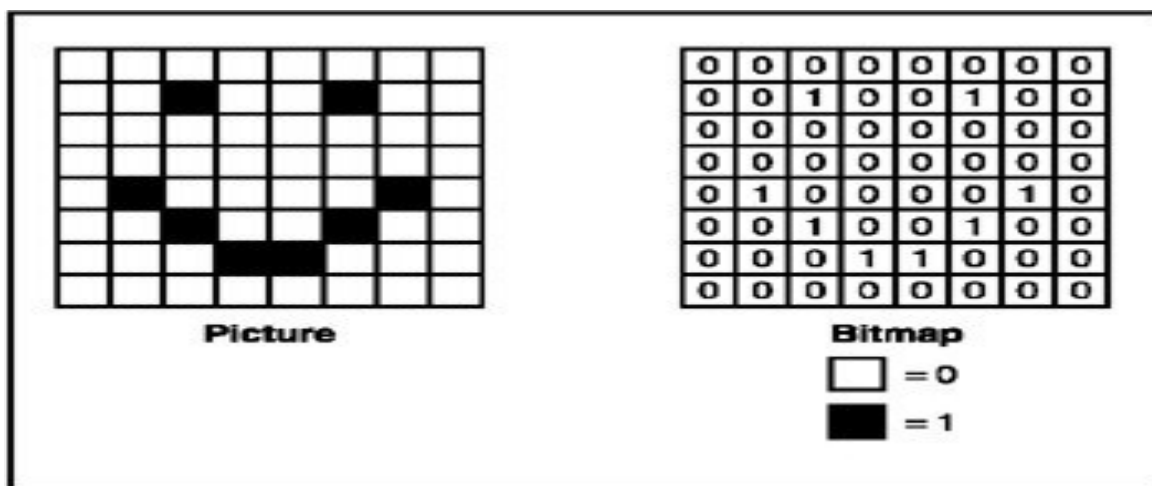
is the foreground color while the rest of the image is the background color. In the document-scanning industry, this is often referred to as "bitonal".

Binary images are also called bi-level or two-level. This means that each pixel is stored as a single bit—i.e., a 0 or 1. The names black-and-white, B&W, monochrome or monochromatic are often used for this concept, but may also designate any images that have only one sample per pixel, such as grayscale images. In Photoshop parlance, a binary image is the same as an image in "Bitmap" mode.

Binary images often arise in digital image processing as masks or as the result of certain operations such as segmentation thresholding, and dithering. Some input/output devices, such as laser printers, fax machines and bilevel computer displays, can only handle bilevel images.

A binary image can be stored in memory as a bitmap, a packed array of bits. A 640×480 image requires 37.5 KiB of storage. Because of the small size of the image files, fax machine and document management solutions usually use this format. Most binary images also compress well with simple run-length compression schemes.

Binary images can be interpreted as subsets of the two-dimensional integer lattice \mathbb{Z}^2 ; the field of morphological image processing was largely inspired by this view.

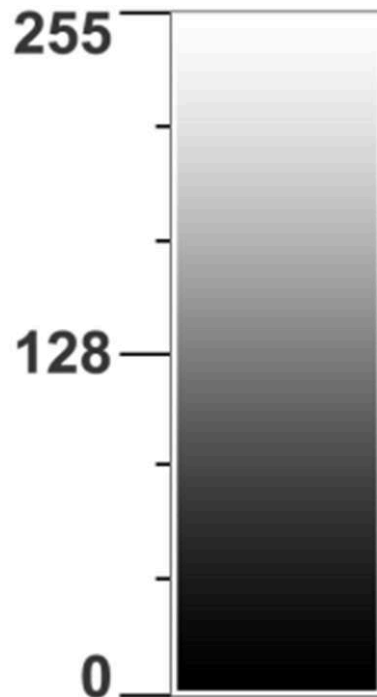


Allowing only one bit per pixel you can create two colors, black and white. These colors will be represented by the binary equivalent. So, for the color white, the binary representation would be '00'. And for Black the binary representation would be '01'.

Quantization and Thresholding

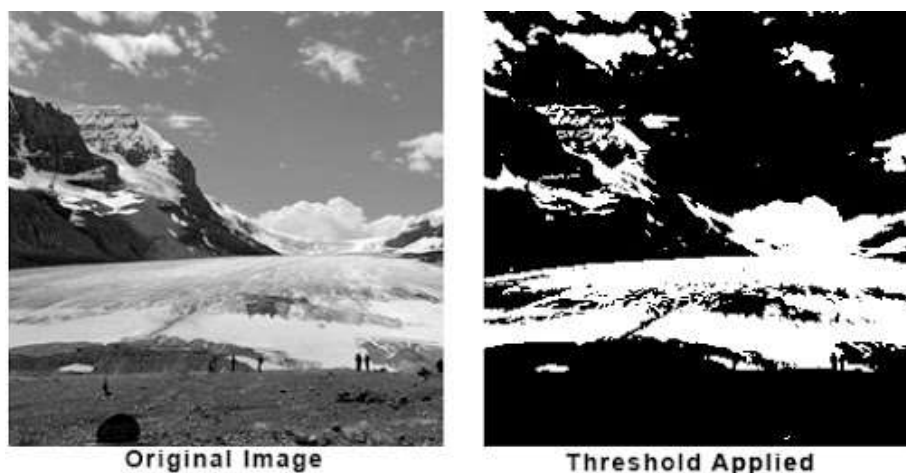
Quantization is the process of converting a continuous range of values into a finite range of discrete values. As the number of bits to represent a pixel intensity (assume Gray scale image for

convenience) is limited, quantization is needed. Suppose 8 bit is used for a pixel, it's equivalent value ranges from 0 to 255 (discrete values). 0 is assigned to pure Black, and 255 is assigned to pure White. Intermediate values are assigned to grey scales as shown in this image. This process is quantization. For 8 bit pixels, the quantization level is 256.



Quantization levels for 8-bit representation of an Image

Image thresholding is a simple, yet effective, way of partitioning an image into a foreground and background. This image analysis technique is a type of image segmentation that isolates objects by converting grayscale images into binary images. Image thresholding is most effective in images with high levels of contrast.



Original Image and its Threshold Image

To improve the threshold image and make it look much clearer, these techniques can be used

1. Halftoning

2. Dithering

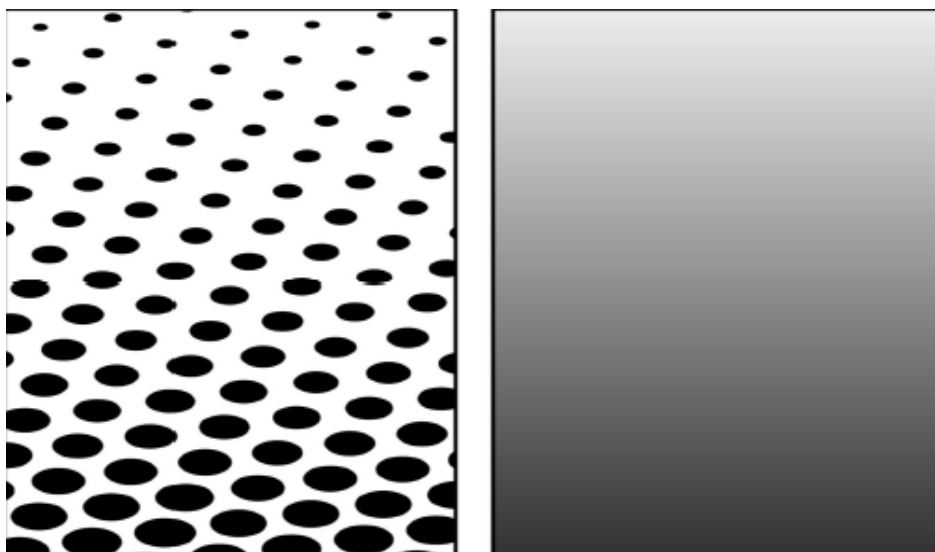
Halftoning

Halftone is the reprographic technique that simulates continuous-tone imagery through the use of dots, varying either in size or in spacing, thus generating a gradient-like effect. "Halftone" can also be used to refer specifically to the image that is produced by this process.

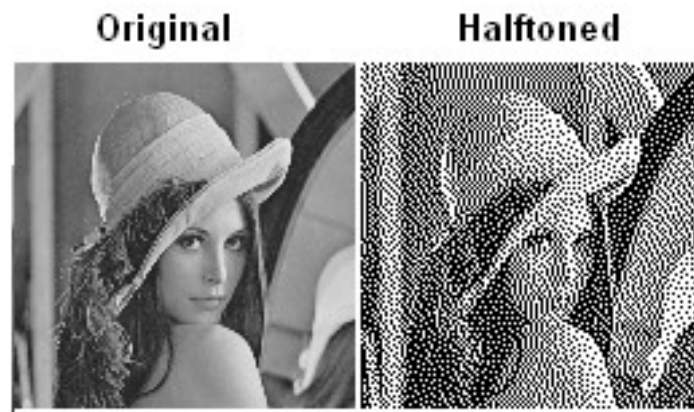
Where continuous-tone imagery contains an infinite range of colours or greys, the halftone process reduces visual reproductions to an image that is printed with only one colour of ink, in dots of differing size (pulse-width modulation) or spacing (frequency modulation) or both. This reproduction relies on a basic optical illusion when the halftone dots are small, the human eye interprets the patterned areas as if they were smooth tones.

At a microscopic level, developed black-and-white photographic film also consists of only two colours, and not an infinite range of continuous tones. For details, see Film grain

Just as colour photography evolved with the addition of filters and film layers, colour printing is made possible by repeating the halftone process for each subtractive colour – most commonly using what is called the "CMYK colour model" The semi-opaque property of ink allows halftone dots of different colours to create another optical effect, full-colour imagery.



Original Halftone Image v/s how a Human Eye sees a Halftoned Image



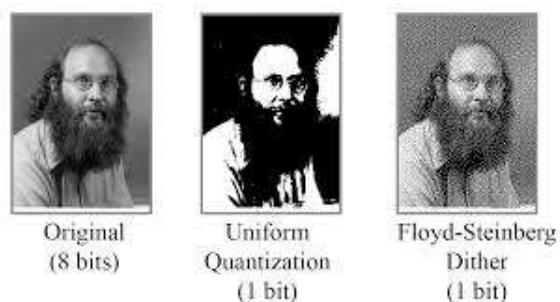
Original Image and Halftoned Image

Dithering

Dithering is used in computer graphics to create the illusion of "color depth" in images with a limited color palette - a technique also known as color quantization. In a dithered image, colors that are not available in the palette are approximated by a diffusion of colored pixels from within the available palette. The human eye perceives the diffusion as a mixture of the colors within it. Dithered images, particularly those with relatively few colors, can often be distinguished by a characteristic graininess or speckled appearance.

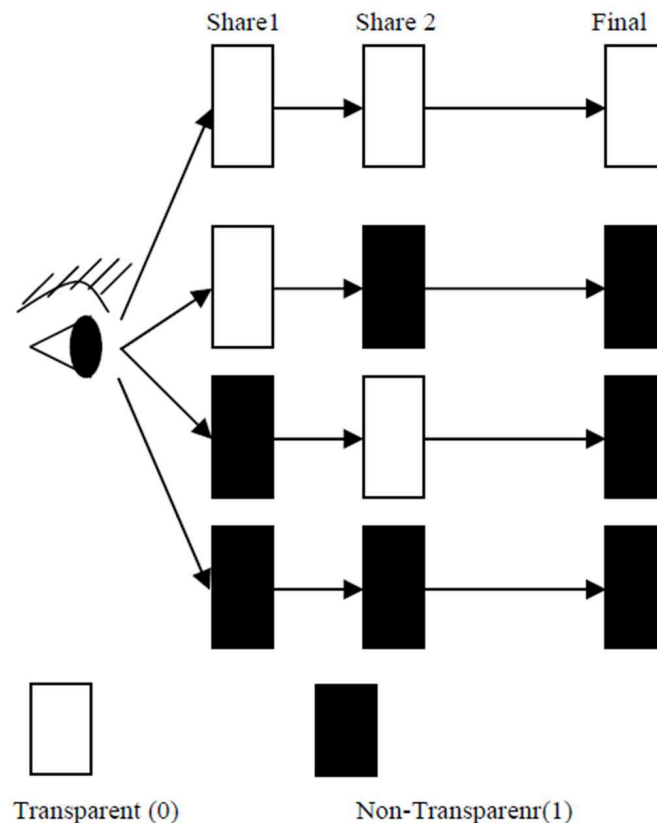
By its nature, dithering introduces pattern into an image - the theory being that the image will be viewed from such a distance that the pattern is not discernible to the human eye. Unfortunately, this is not always the case, and often the patterning is visible - for example, with some images found on the web. In these circumstances it has been shown that a blue-noise dither pattern is the least unsightly and distracting. The error diffusion techniques were some of the first methods to generate blue-noise dithering patterns.

However, other techniques such as ordered dithering can also generate blue-noise dithering without the tendency to degenerate into areas with artifacts.



Original Image, Image with Uniform Quantization and Dithering

Human Visual System



Human Visual System as OR function

Human visual system acts as an OR function. If two transparent objects are stacked together, the final stack of objects will be transparent. But if any of them is nontransparent, then the final stack of objects will be nontransparent. Like OR, $0 \text{ OR } 0 = 0$, considering 0 as transparent and $1 \text{ OR } 0 = 1$, $0 \text{ OR } 1 = 1$, $1 \text{ OR } 1 = 1$, considering 1 as nontransparent.

What is Steganography?

Steganography is the practice of representing information within another message or physical object, in such a manner that the presence of the information is not evident to human inspection. In computing/electronic contexts, a computer file, message, image, or video is concealed within another file, message, image, or video. Steganography includes the concealment of information within computer files. In digital steganography, electronic communications may include steganographic coding inside of a transport layer, such as a document file, image file, program, or protocol. Media files are ideal for steganographic transmission because of their large size. For example, a sender might start with an innocuous image file and adjust the color of every hundredth pixel to correspond to a letter in the alphabet. The change is so subtle that someone who is not specifically looking for it is unlikely to notice the change.

Steganography vs. Cryptography:

Steganography:

Objective: Conceals the existence of a message within a carrier medium.

Method: Embeds information subtly to prevent detection.

Focus: Secrecy through obscurity.

Cryptography:

Objective: Secures the content of a message using mathematical algorithms.

Method: Transforms data using keys for confidentiality.

Focus: Ensures privacy and integrity of the information.

Comparison:

Steganography hides the existence, while cryptography secures the content.

Steganography relies on secrecy, and cryptography on complexity.

Both can be combined for a comprehensive approach to information security.

LSB IMAGE STEGANOGRAPHY

LSB Steganography is an image steganography technique in which messages are hidden inside an image by replacing each pixel's least significant bit with the bits of the message to be hidden.

To understand better, let's consider a digital image to be a 2D array of pixels. Each pixel contains values depending on its type and depth. We will consider the most widely used modes — RGB(3x8-bit pixels, true-color) and RGBA(4x8-bit pixels, true-color with transparency mask). These values range from 0–255, (8-bit values).

We can convert the message into decimal values and then into binary, by using the ASCII Table. Then, we iterate over the pixel values one by one, after converting them to binary, we replace each least significant bit with that message bits in a sequence.

To decode an encoded image, we simply reverse the process. Collect and store the last bits of each pixel then split them into groups of 8 and convert it back to ASCII characters to get the hidden message.

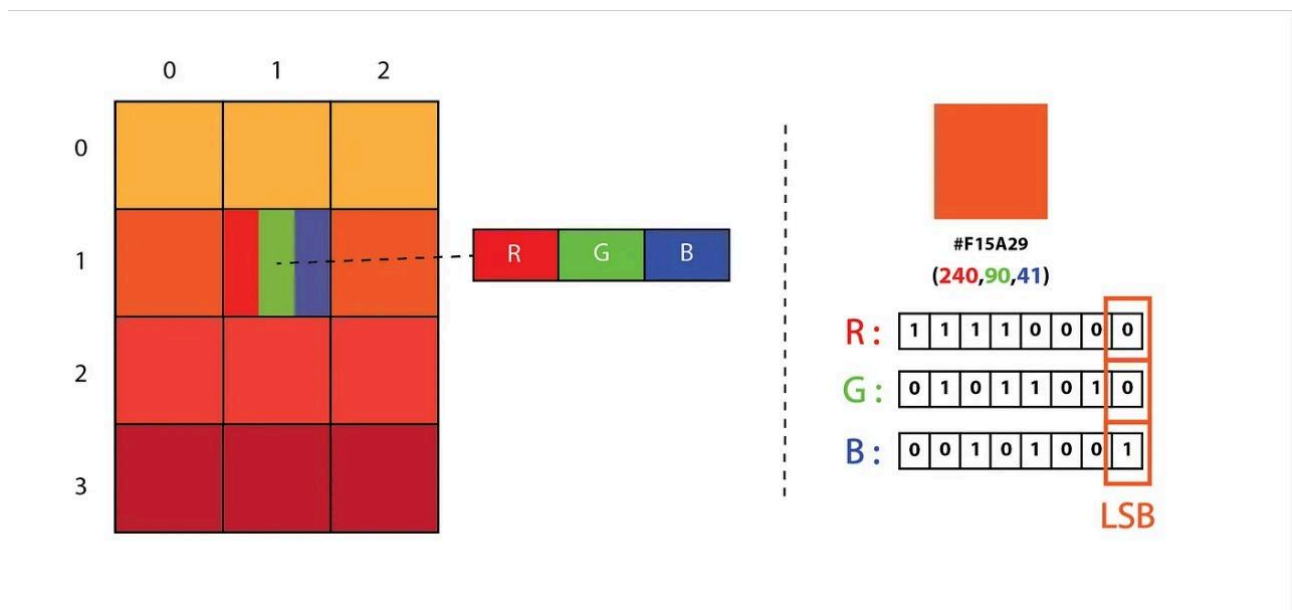


Figure 1.2 LSB Steganography

Steganalysis and Detection Methods:

Steganalysis involves spotting hidden data in stego-images. Techniques include statistical and frequency analysis, and machine learning. Countermeasures involve algorithmic enhancements, encryption integration, and dynamic payload adjustment for added security.

Applications of LSB Steganography:

LSB Steganography finds use in covert communication, securing data transmission, digital watermarking for copyright protection, and forensic applications for identifying image tampering.

Advanced Topics and Research Directions:

Research focuses on robust steganography against attacks, extending steganography to audio and video formats, and integrating it with other security techniques like encryption for comprehensive data protection.

STREAMLIT

Streamlit is an open source app framework in Python language. It helps us create web apps for data science and machine learning in a short time. It is compatible with major Python libraries such as scikit-learn, Keras, PyTorch, SymPy(latex), NumPy, pandas, Matplotlib etc. With Streamlit, no callbacks are needed since widgets are treated as variables. Data caching simplifies and speeds up computation pipelines. Streamlit watches for changes on updates of the linked Git repository and the

application will be deployed automatically in the shared link. It allows developers and data scientists to build interactive and customizable web interfaces using only Python code.

Simplified Web App Development:

Ease of Use: Streamlit is designed for simplicity. Developers can create web applications with just a few lines of Python code, making it accessible to both beginners and experienced programmers.

Rapid Prototyping: Streamlit enables quick prototyping of data-driven applications, reducing the time and effort needed for development.

Data Visualization:

Built-in Components: Streamlit provides a variety of built-in components for creating charts, plots, and interactive visualizations without the need for extensive coding.

Integration with Plotting Libraries: It seamlessly integrates with popular Python plotting libraries such as Matplotlib, Plotly, and Altair.

Interactive Widgets:

User Input Elements: Developers can easily add interactive widgets, such as sliders, buttons, and text inputs, allowing users to control and customize the application.

Real-Time Updates: Widgets can trigger real-time updates to the displayed content, enabling dynamic and interactive user experiences.

Deployment and Sharing:

Single-Command Deployment: Streamlit apps can be deployed with a single command, making it straightforward to share applications with others.

Compatibility: Streamlit apps can be deployed on various platforms, including cloud services, allowing for easy accessibility.

Customization and Theming:

Custom Components: Developers can create custom components for additional functionality tailored to their specific needs.

Theming: Streamlit supports theming options, allowing developers to customize the appearance of their applications.

Integration with Machine Learning:

Seamless Integration: Streamlit is often used in conjunction with popular machine learning libraries like TensorFlow and PyTorch to create interactive dashboards showcasing model outputs.

Open Source Community:

Active Community: Streamlit has a vibrant open-source community that contributes to its development and provides support through forums and documentation.

Supported File Types:

Image and Video Integration: Streamlit supports the integration of images and videos, enhancing the multimedia capabilities of web applications.

1.2 PROBLEM STATEMENT

Let 'D' be the secret image to be shared among 'n' parties. A (k, n)-threshold scheme is a way to divide 'D' into 'n' pieces D_1, \dots, D_n that satisfies the following conditions:

1. Knowledge of any 'k' or more D_i pieces makes 'D' easily computable,
2. Knowledge of any 'k - 1' or fewer D_i pieces leaves 'D' completely undetermined (in the sense that all its possible values are equally likely).

We can use this process to share confidential image data over a channel to different stakeholders. Suppose we have to send confidential data to be shared among a group of people. We can split the image into 'n' shares and make it compulsory for at least 'k' images to be merged to get the original image back. By doing so we can send 'n' shares of a secret image to 'n' stakeholders. They can only access this confidential information only if at least 'k' stakeholders agree to access it.

There have to be at least 'k' images to get back the original image. By doing so we can provide a high level of collaborative security.

We also maintain a secret key value to provide extra security that will encrypt the original image. So, now to get the original image we not only need 'k' image shares but also the key to decrypt it. Now it becomes very difficult to access the image by an unauthorized person.

1.3 SIGNIFICANCE OF THE PROBLEM

The (k, n)-threshold scheme addresses a critical need in collaborating secret sharing. By dividing the image into shares that require cooperation from a minimum threshold of authorized stakeholders to reconstruct, the scheme ensures resilience against unauthorized access. Integration of encryption with a secret key adds an extra layer of security, enhancing the protection of the information. The

scheme significantly contributes to maintaining the confidentiality and security of vital collaborative assets.

1.4 BRIEF DESCRIPTION OF SOLUTION APPROACH

The proposed solution approach begins with selecting a secret image for secure transmission. This image undergoes encryption using both symmetric (e.g., Caesar Cipher) and asymmetric (e.g., RSA) key encryption techniques. Subsequently, the encrypted image is divided into 'n' shares using a (k, n)-threshold secret sharing algorithm, with a random number generator determining the shares and a reconstruction factor of (n-k+1). Each share is then distributed to 'n' different individuals via mail by the admin. To reconstruct the original image, at least 'k' shareholders must agree to combine their shares, sending them back to the admin. Upon receiving the shares, the admin combines them and decrypts the resultant image using RSA and Caesar Cipher decryption algorithms with the proper keys. If the minimum share requirement and correct key usage are met, the admin obtains the secret image, which can be accessed by the consenting shareholders directly from the admin's system.

1.5 COMPARISON OF EXISTING APPROACH TO PROBLEM FRAMED

Existing Approaches	Technique	Weakness	How Secured Palette Overcomes Weakness
Traditional Encryption Methods	Using traditional encryption algorithms like AES, DES	Vulnerable to brute force attacks, single point of failure if key compromised	Our project combines multiple encryption techniques (e.g., Caesar cipher, RSA) along with image steganography for added security. This multi-layered approach makes it significantly harder for attackers to

			decrypt the image, even if one layer is compromised.
Simple Image Sharing	Dividing image into equal parts and distributing among recipients	Lack of threshold security, vulnerable to collusion attacks	Our project implements a (k, n) -threshold scheme where knowledge of any 'k' or more shares is required to reconstruct the original image. Even if some shares are compromised, the original image remains secure unless a threshold number of shares are collected.
Basic Image Steganography	Hiding data within image pixels using LSB method	Limited security, susceptible to detection and removal	Our project enhances image steganography with encryption techniques and threshold security. By encrypting the hidden data and requiring a threshold number of shares for decryption, it adds an extra layer of security, making it more resilient against detection and removal attacks.

Secret sharing schemes	Dividing data into multiple shares for sharing and reconstruction	Can be complex to manage and requires multiple shares	Our project employs visual cryptography, eliminating the need to manage multiple shares and enabling data reconstruction with just two shares.
Error-correcting codes	Embedding data in the redundancy added to the code for error protection	Can be complex to implement	Secured Palette's LSB steganography method is inherently self-correcting in case of minor errors, reducing the complexity and improving robustness.
Information dispersal algorithms	Dividing data into multiple shares for resilient data hiding	Can be complex to implement	Secured Palette does not require the implementation of complex algorithms, making it more accessible and user-friendly.
Spread spectrum modulation	Embedding data in the signal's frequency spectrum	Can be complex to implement	Secured Palette's LSB steganography method is simpler to implement compared to spread spectrum modulation, offering a balance between security and practicality.

Covert channels	Embedding data in the timing or structure of communication channels	Can be unreliable and susceptible to detection	Secured Palette utilizes robust steganography and visual cryptography techniques, making it less reliant on unreliable covert channels.
Zero-steganography	Exploiting limitations of data hiding algorithms for secure data embedding	Can be difficult to implement and requires careful design	Secured Palette's combination of LSB steganography and visual cryptography provides a more straightforward and practical approach to secure data embedding.

Table 1.1: comparison of existing approach to problem framed

CHAPTER-2

LITERATURE SURVEY

2.1 LITERATURE SURVEY

VISUAL CRYPTOGRAPHY

1. Naor, Moni, and Adi Shamir. "Visual cryptography." In *Advances in Cryptology—EUROCRYPT'94: Workshop on the Theory and Application of Cryptographic Techniques* Perugia, Italy, May 9–12, 1994 Proceedings 13, pp. 1-12. Springer Berlin Heidelberg, 1995.

The paper introduces a groundbreaking cryptographic scheme designed for the easy decryption of concealed images without the need for complex cryptographic computations. This visual cryptography approach combines a printed page of ciphertext with a printed transparency, serving as a secret key, allowing the original content to become visible when the two are stacked together, even though both initially appear as random noise

2. Pawar, Shital B., and N. M. Shahane. "Visual Secret Sharing Using Cryptography." *International Journal of Engineering Research* 3, no. 1 (2014): 31-33.

The paper discusses the Embedded Extended Visual Cryptography Scheme (EEVCS), which is a variation of the Visual Cryptography Scheme (VCS) used for sharing secret images. In VCS, a secret image is split into random shares, and the secret can be reconstructed by stacking these shares using logical OR operations. EEVCS enhances VCS by using meaningful covering shares, achieved by embedding random shares of the secret image into cover images.

3. Pandey, Anjney, and Subhranil Som. "Applications and usage of visual cryptography: A review." In *2016 5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO)*, pp. 375-381. IEEE, 2016.

The paper discusses the importance of data security and encryption in today's computer generation, particularly in light of the rise in cybercrime. Visual Cryptography is highlighted as a significant part of cryptography, with various application areas, including biometric security, watermarking, remote electronic voting, and bank customer identification.

4. K. Shankar and P. Eswaran, "A new k out of n secret image sharing scheme in visual cryptography," Jan. 2016.

The paper has presented the analysis of Otsus threshold method and LSB matching steganography algorithm for addressing the challenging problem of visual quality of embedded shares in cryptography based visual secret sharing system. The method proposed in the paper not only increases the visual quality of recovered secret but also gives better results. It also improves the PSNR as compared to other methods.

IMAGE STEGANOGRAPHY

5. Subramanian, Nandhini, Omar Elharrouss, Somaya Al-Maadeed, and Ahmed Bouridane. "Image steganography: A review of the recent advances." IEEE access 9 (2021): 23409-23423.

The research paper, "Image Steganography: A Review of the Recent Advances," delves into the realm of multimedia data security with a particular emphasis on steganography and cryptography in the context of the Internet of Things (IoT). Focusing on conventional encryption for data security, the paper also introduces steganography as a method to clandestinely embed information within multimedia content. The integration of steganography and cryptography within the IoT framework is explored, showcasing the paper's comprehensive review of recent advances in image steganography.

6. Mstafa, Ramadhan J. "Information hiding in images using steganography techniques." ASEE, 2013.

The research paper, "Information Hiding in Images Using Steganography Techniques," addresses privacy concerns arising from the widespread distribution of information facilitated by technological advancements. The paper specifically concentrates on the role of steganography in mitigating these concerns by safeguarding sensitive data. Steganography is explored as a technique to conceal information within images, offering a method to protect data by embedding it within seemingly innocuous visual content. The focus on information hiding within images signifies the paper's contribution to enhancing privacy and security measures in the context of global information dissemination.

7. Somwanshi, D. R., and V. T. Humbe. "Half-Tone Visual Cryptography Scheme For RGB Color Images." Indian Journal of Science and Technology 16, no. 5 (2023): 357-366.

The linked paper titled "Half-Tone Visual Cryptography Scheme for RGB Color Images" explores a novel approach to Visual Cryptography tailored for RGB color images. Visual Cryptography involves dividing images into shares, and this paper focuses on adapting the technique to handle the complexities of color images represented in the RGB model. The term "Half-Tone" suggests a

method involving the use of halftone patterns, commonly used in printing, to enhance the security of the visual cryptography scheme. By employing this approach, the research likely aims to provide a secure and efficient means of sharing and reconstructing RGB color images through visual cryptography.

8. Cheddad, Abbas, Joan Condell, Kevin Curran, and Paul Mc Kevitt. "Digital image steganography: Survey and analysis of current methods." *Signal processing* 90, no. 3 (2010): 727-752.

The paper, "Digital Image Steganography," delves into the realm of concealing information within digital images using steganographic techniques. Digital image steganography involves hiding data within the pixels of an image without perceptibly altering its visual appearance. The paper explores various steganographic methods, their applications, and potential implications for information security.

9. Gupta, Ravindra, Akanksha Jain, and Gajendra Singh. "Combine use of steganography and visual cryptography for secured data hiding in computer forensics." *International Journal of Computer Science and Information Technologies* 3, no. 3 (2012): 4366-4370.

The paper encodes the secret message in at least significant bits of the original image, where the pixels values of the encrypted image are modified by the genetic algorithm to retain their statistic characters, thereby making the detection of secret of message difficult. Use of Genetic algorithm has compelled the system for enhancing the security. The proposed system hides data in a real image and achieve its detection after under went to visual cryptography. The main aim of the proposed model is to design a feasible RS- resistance secure algorithm which combines the use of both steganography and visual cryptography for improving security, reliability and efficiency for secret message.

2.2 INTEGRATED SUMMARY OF LITERATURE STUDIED

The collective insights from these research papers align with the core principles of our project, which utilizes a combination of LSB Steganography and Visual Cryptography. The RGB-Based Secret Sharing Scheme in Color Visual Cryptography resonates with our project's emphasis on securing information transmission, especially in the context of color images.

Additionally, the investigation into privacy concerns and the role of Steganography in safeguarding sensitive data aligns with our project's overarching goal of concealing information within images for enhanced privacy. The proposal of a Half-Tone Visual Cryptography Scheme for RGB color images

further contributes to our project's exploration of innovative approaches to secure Visual Cryptography, especially in the context of color images.

Furthermore, the collaborative research on combining Steganography and k-n shares Visual Cryptography for secure data hiding in computer forensics echoes the practical integration aspect of our project. It emphasizes the real-world applicability of these techniques, aligning with our project's objective of providing a robust solution for secure information sharing and protection. In summary, the insights from these research papers complement and validate the methodologies employed in our project, reinforcing the relevance and effectiveness of combining LSB Steganography and k-n share Visual Cryptography for enhanced data security and confidentiality.

CHAPTER-3

REQUIREMENT ANALYSIS AND SOLUTION APPROACH

3.1 OVERALL DESCRIPTION OF THE PROJECT

The project entails the secure transmission of a secret image through a series of encryption and decryption processes. Beginning with encryption using Caesar cipher and RSA techniques, the encrypted image is split into shares using secret sharing algorithms, distributed to recipients via mail, and merged securely for decryption. The final decrypted image is obtained through a meticulously orchestrated process, ensuring confidentiality and integrity throughout.

IMAGE SELECTION:

An image to be secretly transmitted is selected.

ENCRYPTION PROCESS:

Caesar Cipher Encryption: The selected image undergoes encryption using the Caesar cipher. A key is utilized to compute a value, which is then added to all the pixels of the image. This resultant value is then modded by 256.

RSA Encryption: Following Caesar cipher encryption, RSA encryption is employed. A pair of public and private keys is computed. The public key is then used to encrypt all the pixel values of the image.

IMAGE SHARING (VISUAL CRYPTOGRAPHY):

The application splits the stego-image (the image carrying the hidden message) into 'n' shares using the (k,n) secret sharing encryption algorithm. Each share is sent to individual recipients via mail.

DECODING PROCESS:

Selection of Shares for Merging: At least 'k' out of the 'n' shares are chosen for merging.

Combining Shares: The selected shares are overlapped and combined using the (k, n) secret sharing decryption algorithm.

DECRYPTION PROCESS:

RSA Decryption: Decryption commences by using RSA. The private key generated earlier is utilized for RSA decryption.

Caesar Cipher Decryption: Subsequently, the key used for Caesar cipher encryption is employed for decryption using Caesar cipher. It's essential to ensure that the order of decryption matches the reverse order of encryption.

FINAL IMAGE RETRIEVAL:

After completing the decryption steps, the final image is obtained.

KEY FEATURES:

- **SECURE DATA HIDING:** Utilizes LSB steganography to embed confidential information within images
- **VISUAL CRYPTOGRAPHY IMPLEMENTATION:** Employs VC to split the stego-image into 'n' shares for secure transmission and reconstruction
- **USER-FRIENDLY WEB INTERFACE:** Provides a simple and intuitive web interface for encoding and decoding messages

3.2 REQUIREMENT ANALYSIS

REQUIREMENT	DESCRIPTION
Streamlit for Web Development	Streamlit, a Python library designed for creating interactive and data-driven web applications, was chosen for its simplicity and ease of use. The Streamlit framework enabled the development of a user-friendly web interface for encoding and decoding messages. Streamlit's built-in functionalities for file uploading, image processing, and data visualization facilitated the implementation of the steganography and visual cryptography algorithms.
Python as the Primary Programming Language	Python was selected as the primary programming language due to its:
Versatility	Python's general-purpose nature made it suitable for various tasks, including image processing, data manipulation, and web development.

Extensive Libraries	Python's rich ecosystem of libraries provided ready-made tools for image processing, cryptography, and web development.
Widespread Community Support	Python's large and active community ensured easy access to support and resources.
Git for Version Control	Git, a distributed version control system, was employed to:
Track changes in the codebase	Git maintained a history of changes, allowing developers to revert to previous versions if necessary.
Manage collaborative development	Git facilitated collaboration among developers by enabling them to merge their changes and work on different parts of the project simultaneously.
Ensure project stability	Git's branching and merging capabilities helped maintain project stability by isolating changes and preventing conflicts.

Table 3.1 Requirement Analysis

3.3 SOLUTION APPROACH

1. ENCRYPTION:

Encryption is that process where the data is transformed into an incomprehensible data that is not in its original format. Encryption is necessary to provide more security to the secret that is being transmitted so that even if the shares are taken by the intruders, they cannot form the secret back to its original form. In this project, we make use of 2 encryption algorithms: first one is a simple and easy symmetric encryption algorithm Caesar Cipher where in every value in each pixel is added with a fixed value(key) and then mod with 256 because every value in a pixel is in the range of 0-255, the next encryption used is a stronger algorithm RSA which makes use of two pairs of keys(public and private).Two encryption algorithms are used to make the security much stronger.

Caesar Cipher:

The Caesar Cipher technique is one of the earliest and simplest method of encryption technique. It's

simply a type of substitution cipher, i.e., each letter of a given text is replaced by a letter some fixed number of positions down the alphabet. For example, with a shift of 1, A would be replaced by B, B would become C, and so on. The method is apparently named after Julius Caesar, who apparently used it to communicate with his officials. Thus, to cipher a given text we need an integer value, known as shift which indicates the number of positions each letter of the text has been moved down. In this project, the key is calculated by adding all the ASCII values of the key(text) that is entered. We get a value here which is added to each of R, G, B values of every pixel and then mod it with 256 since a value in each pixel would be in the range of 0, 1,, 255. By now we will have an image that is encrypted by Caesar Cipher. Since this algorithm is easy to crack, we make use of RSA as well.

RSA:

RSA is an algorithm used by modern computers to encrypt and decrypt messages. It is an asymmetric cryptographic algorithm, meaning there are two different keys. This is also called public key cryptography because one of them is public i.e., can be given to everyone. The other key must be kept private. It is based on the fact that finding the factors of an integer is hard. RSA stands for Ron Rivest , Adi Shamir and Leonard Adleman , who first publicly described it in 1978. RSA involves a public key and private key. The public key can be known to everyone, it is used to encrypt messages. Messages encrypted using the public key can only be decrypted with the private key.

2. GENERATE AND SEND EMAILS TO EACH SHAREHOLDER

Emails are sent to user's emails provided by the admin. Here each user would be a stakeholder. The number of emails is equal to the number of shares created in the first place. Now, each user will get a share of the secret image.

E-mails are sent directly by a python code by making use of the modules smtplib, MIMEMultipart, MIMEText, MIMEBase and by entering the e-mail ids of each shareholder after those shares are generated.

3. OVERLAPPING 'k' SHARES (MERGING 'k' SHARES)

To reconstruct the image from n shares, at least k shares of those n shares are definitely required. Even though the knowledge of k-1 shares cannot form the original image. Now if at least k out of n shareholders accept to contribute their shares, then the image can be reconstructed. Image is constructed from these shares based on OR operation so that every non-zero pixel is considered and used for image construction. As a result, the image is ready for decryption.

4. DECRYPTION

The image reconstructed from shares is decrypted in this module. The reconstructed image is first

decrypted using the RSA algorithm. Now the resultant is then decrypted with a key using Caesar Cipher. These keys must be the same or related to the ones that are used during encryption. Wrong key won't give us the original image back and the secret won't be revealed properly.

The order in which we use the decryption algorithms is very important. Since we encrypted the image first by Caesar Cipher and then by RSA, in the decryption process we should first decrypt the image by using RSA decryption algorithm and by then that of Caesar Cipher. If the order of encryption and decryption aren't exactly opposite of each other, the final image will not be the same as the original image. Also, the keys used to decrypt the image have to be exactly correct to get a correct image by the end.

CHAPTER-4

MODELING AND IMPLEMENTATION DETAILS

4.1 DESIGN DIAGRAMS

4.1.1 USE CASE DIAGRAM

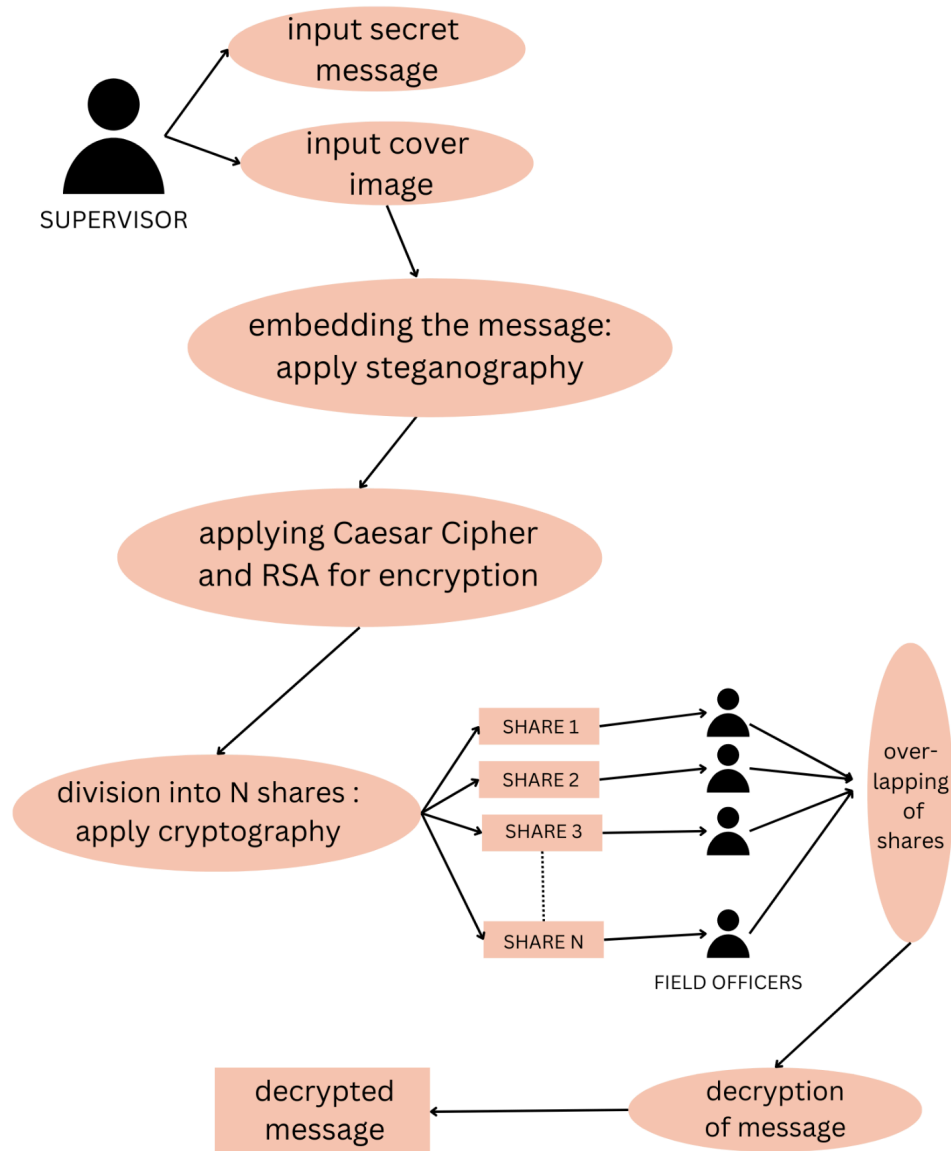
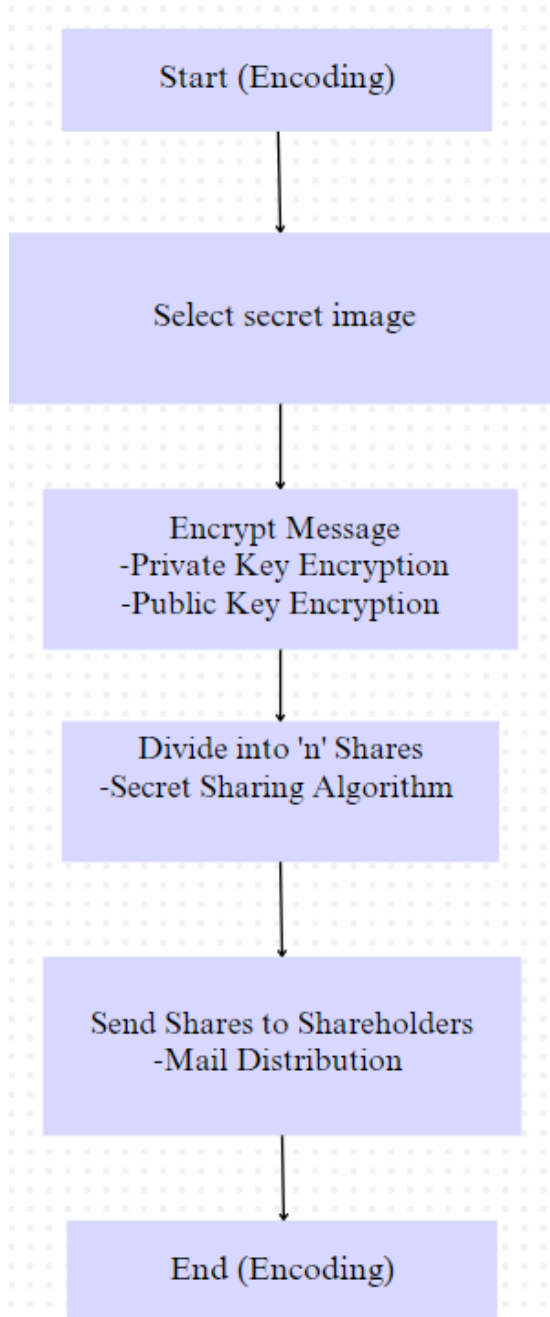


Figure 4.1: use case diagram

4.1.2 CONTROL FLOW DIAGRAMS

ENCODING FLOW



DECODING FLOW

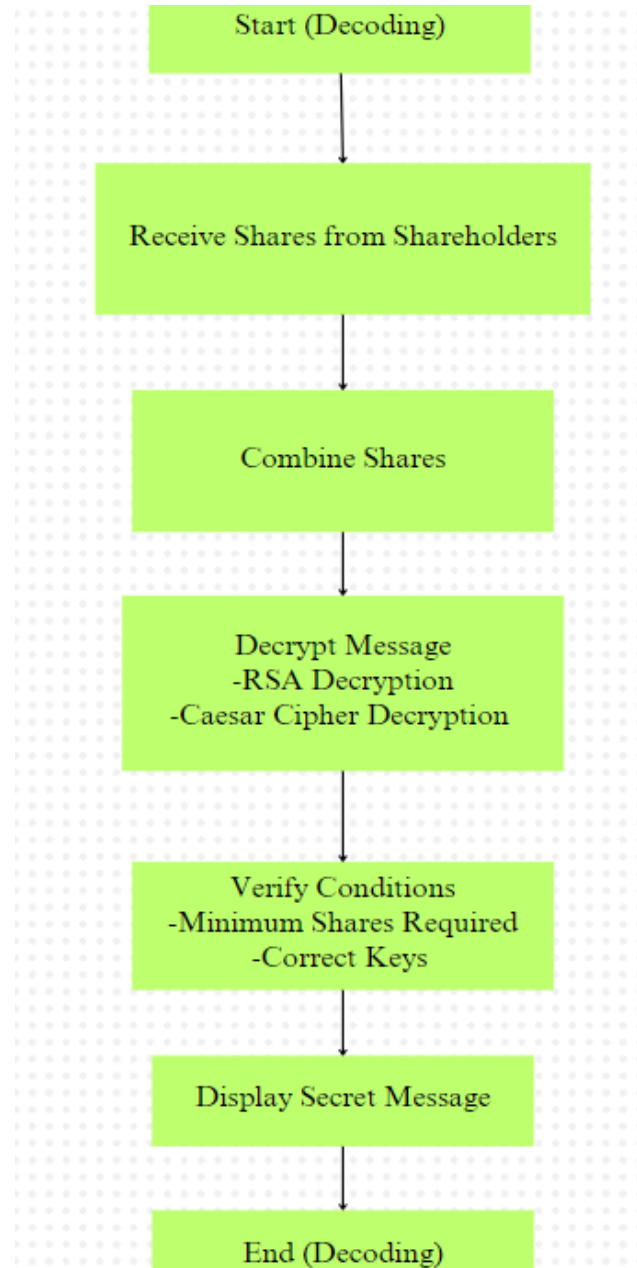


Figure 4.2: use case diagram - encoding Figure 4.3: use case diagram - decoding

4.1.3 CLASS DIAGRAM

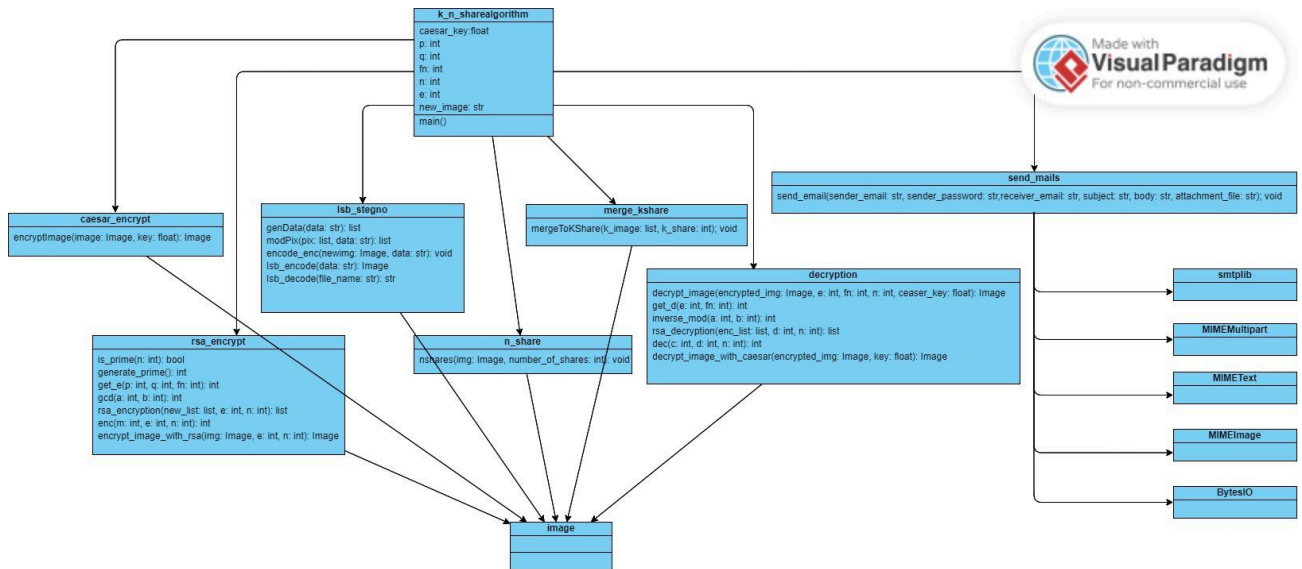


Figure 4.4: class diagram

4.1.4 ACTIVITY DIAGRAM

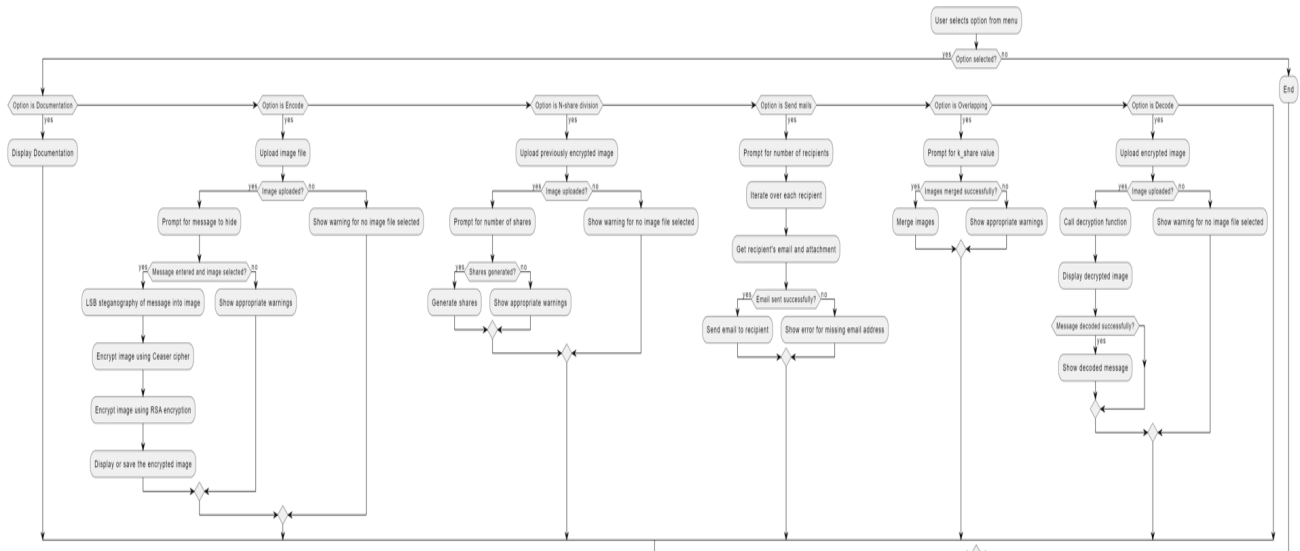


Figure 4.5: activity diagram

4.2 IMPLEMENTATION DETAILS AND ISSUES

IMPLEMENTATION DETAILS

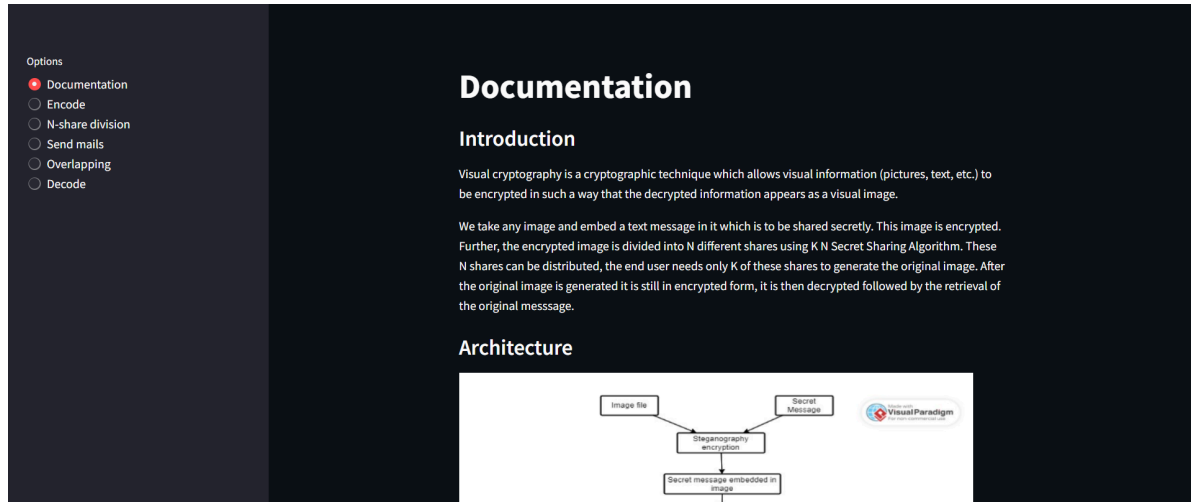


Figure 4.6: Explaining the principles of k - n sharing visual cryptography algorithm and providing details on control flow of the project.

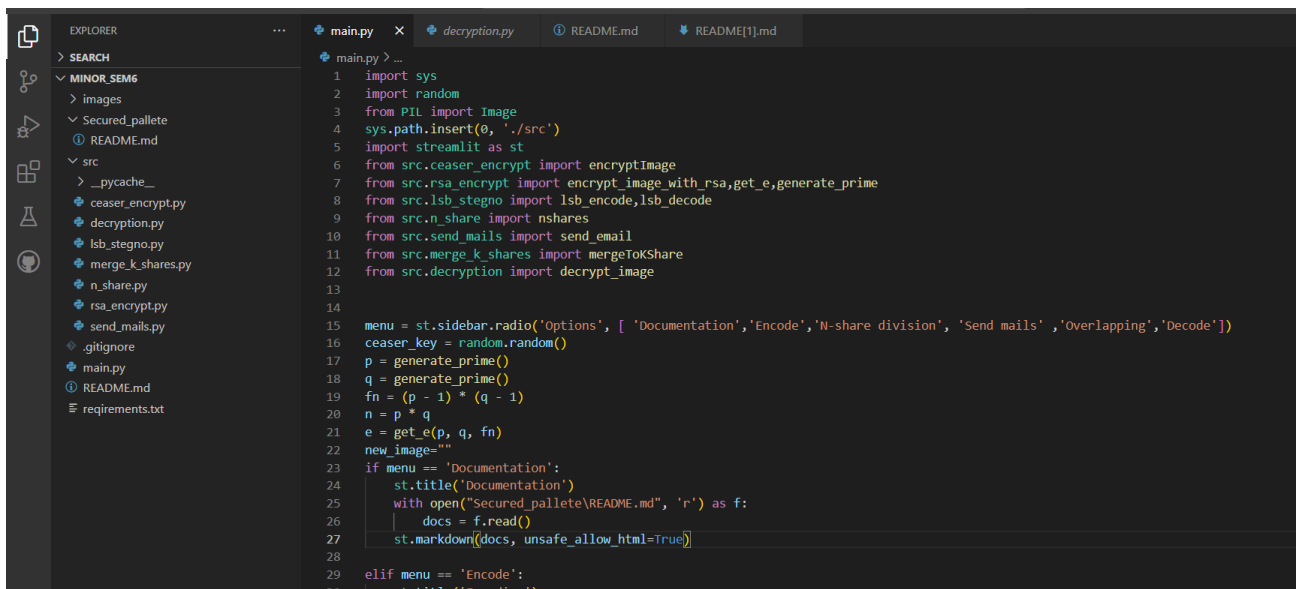


Figure 4.7: Streamlit Integration: The project leverages the Streamlit framework to create a dynamic and interactive web interface.

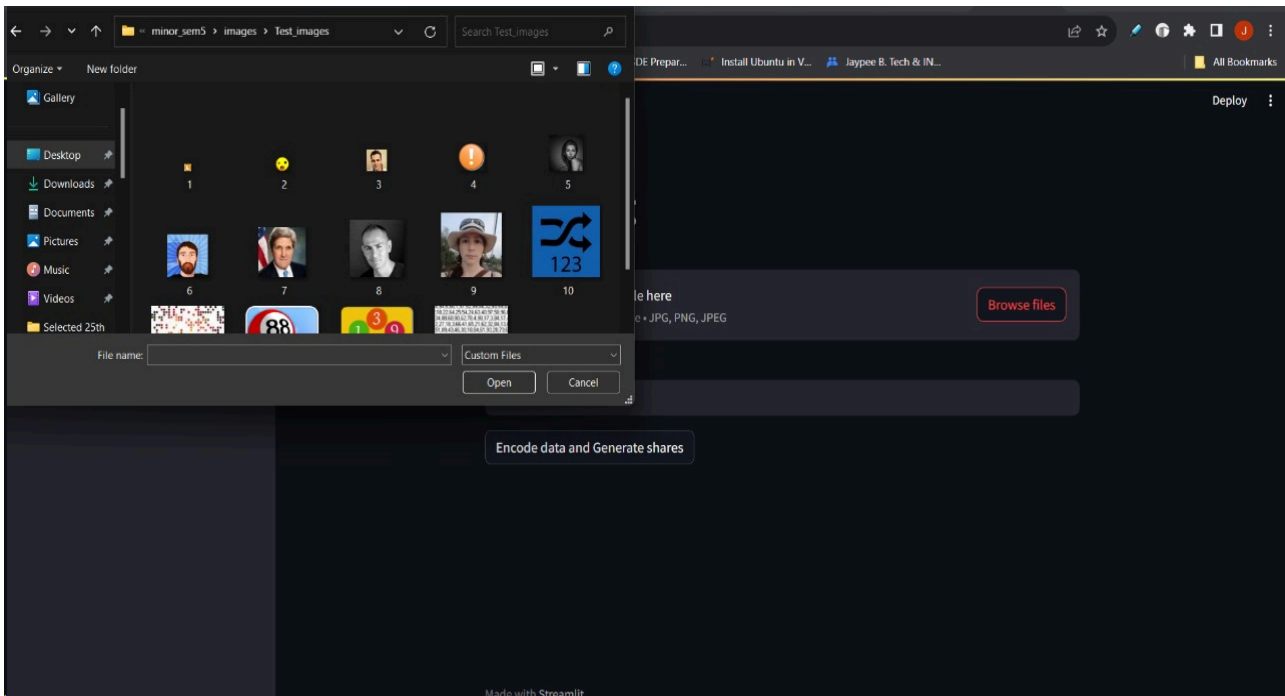


Figure 4.8: Image Selection: Users can choose any image from their local device to serve as the cover for hiding the secret message. A user-friendly interface allows for easy navigation and selection of images.

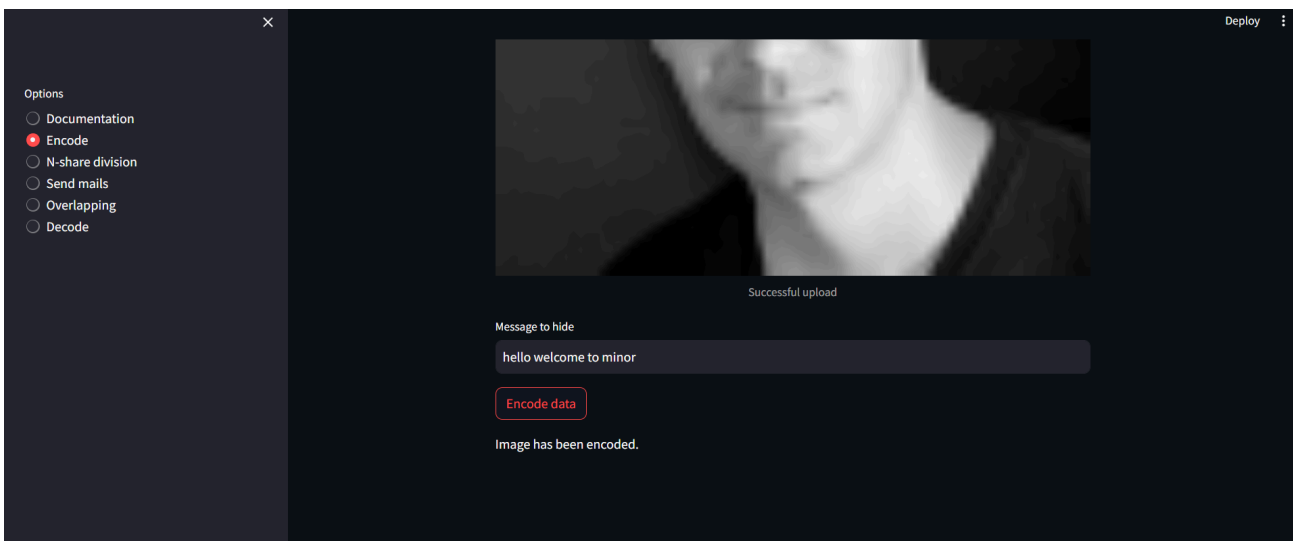


Figure 4.9: Message Input: The web interface includes a text input field where users can type the secret message they want to embed in the selected image. Also the image is encoded and saved into the system.

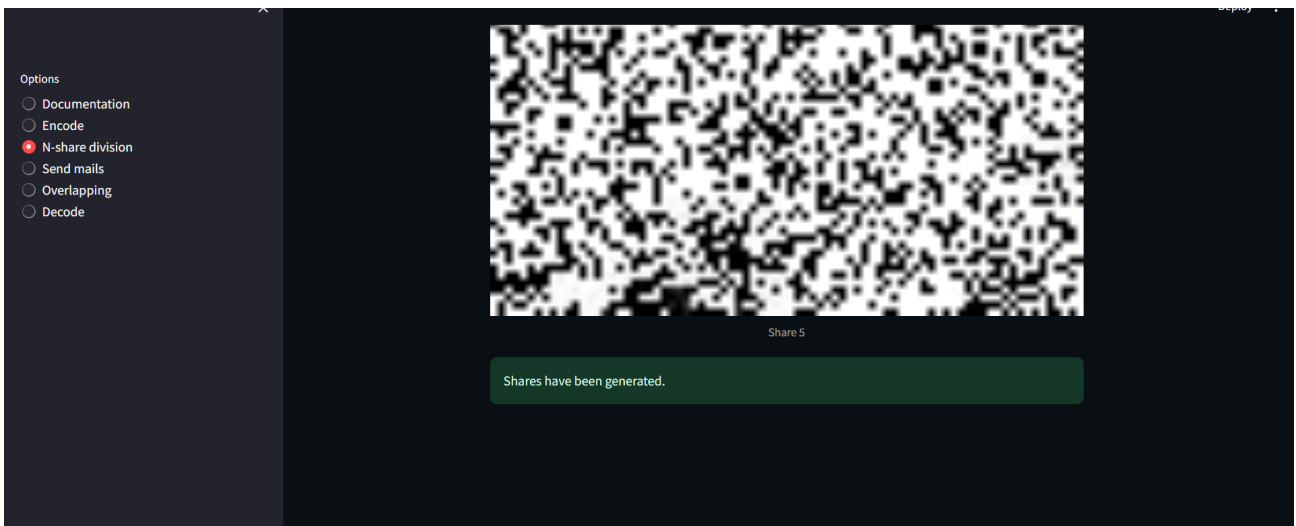


Figure 4.10:Share Generation: The web interface includes a number input field where users can enter the number of shares to be generated and the shares are saved and displayed on the screen.

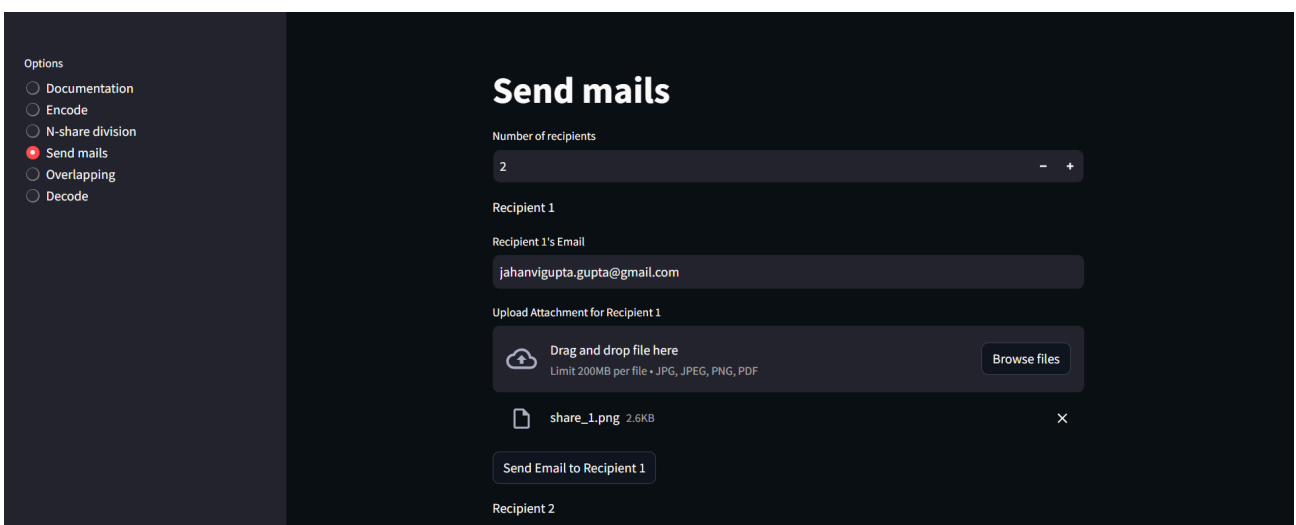


Figure 4.11:Send mails: Users can upload the generated shares of the encoded image through the web interface and the admin will enter the recipient's email id.

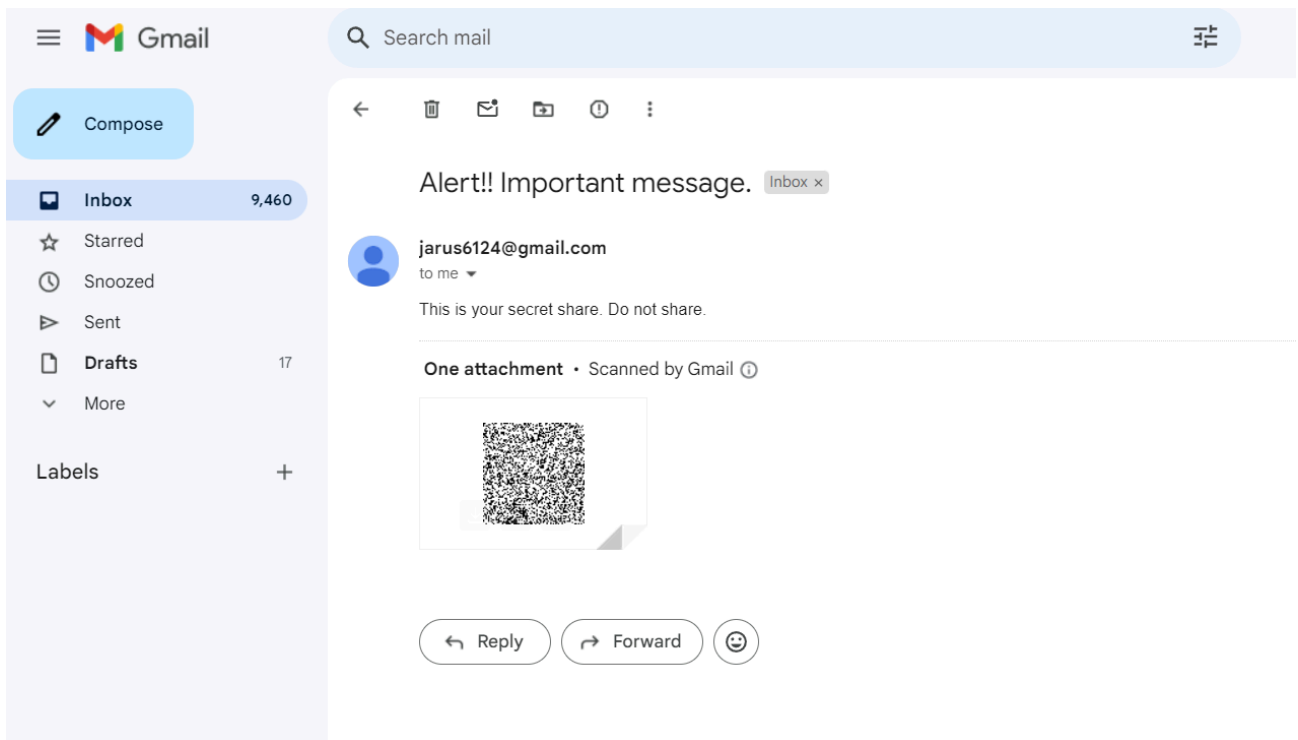


Figure 4.12: Mail received by each recipient

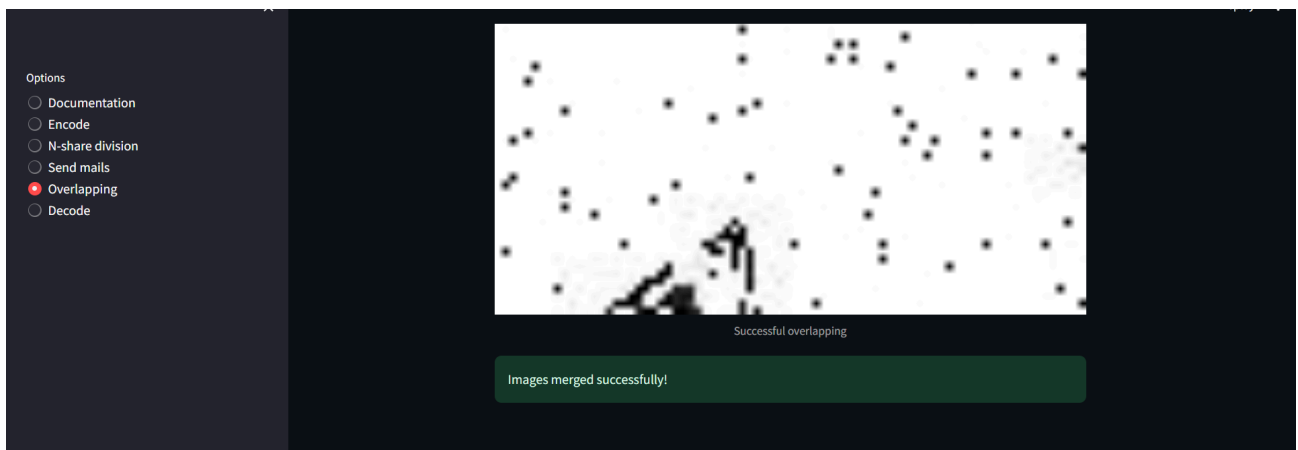


Figure 4.13: Share overlapping: user will enter the value of k and the algorithm will produce an overlapped image of the shares.

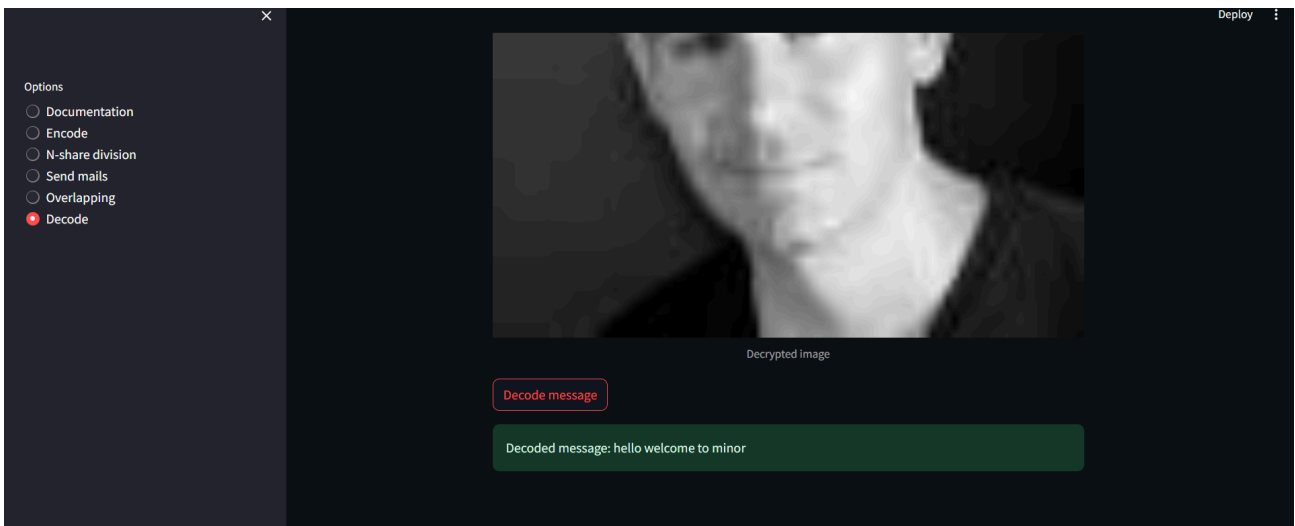


Figure 4.14:Decryption: user will input the overlapped image produced earlier and the algorithm will produce a decrypted image and the message will be revealed.

ALGORITHMS

Algorithm: RSA

1. Choose two distinctive prime numbers p and q . For security purposes, the integers p and q should be chosen at random and should be of similar bit length. Prime integers can be efficiently found using a primality test.
2. Compute $n = p * q$
 n is used as the modulus for both the public and private keys. It's length, usually expressed in bits, is the key length.
3. Compute $\Phi(n) = \Phi(p) * \Phi(q)$
 $= (p - 1) * (q - 1)$
 $= n - (p + q - 1)$
 where, $\Phi(n)$ is Euler's totient function.
4. Choose an integer e such that, $1 < e < \Phi(n)$ and $\gcd(e, \Phi(n)) = 1$ that is e and $\Phi(n)$ are coprime.
5. Determine d as $d = e^{-1} \pmod{\Phi(n)}$ i.e., d is the multiplicative inverse of e (modulo $\Phi(n)$).

Algorithm: Division into 'n' shares

1. Take an image as input and calculate its width (w) and height (h).
2. Take the number of shares (n) and minimum number of shares (k) to be taken to reconstruct the image. k must be less than or equal to n .
3. Calculate $\text{recons} = (n - k) + 1$.

4. Create a three dimensional array `img_share[n][w*h][32]` to store the pixels of n number of shares.

5.

for `i=0` to `(w*h-1)`

{

Scan each pixel value of the image and convert it into 32 bit binary string let PIX.

for `j=0` to `31`

{

if ith position of PIX contains '1' call `Random_Place(n, recons)`

for `k=0` to `(recons-1)`

{

Set `img_share[rand[k]][i][j] = 1`

}

}

}

6. Create a one dimensional array `img_cons[n]` to store constructed pixels of each share.

7.

for `k1=0` to `(n-1)`

{

for `k2=0` to `(w*h-1)`

String value= ""

for `k3=0` to `31`

{

`value=value+img_share[k1][k2][k3]`

}

construct alpha, red, green and blue part of each pixel by taking consecutive 8 bit substring starting from 0. Construct pixel from these part and store it into `img_cons[k1]`

}

generate image from `img_cons[k1]`

}

subroutine int Random_Place(n,recons)

{

create an array `rand[recons]` to store the random number generated.

for `i=0` to `(recons-1)`

{

```

generate a random number within n, let rand_int if(rand_int is not in rand[recons])
rand[i] = rand_int.
}
return rand[recons]
}

```

4.3 RISK ANALYSIS AND MITIGATION

Risk	Description	Likelihood	Impact	Mitigation Strategy
Data loss	The image and/or text message could be lost during the encoding or decoding process.	Medium	High	Implement data redundancy and error-correction techniques to ensure data integrity.
Unauthorized access	The image and/or text message could be accessed by unauthorized individuals.	Low	Medium	Implement strong encryption and access control mechanisms to protect sensitive data.
Steganalysis	The presence of the hidden message could be detected using steganalysis techniques.	Low	Medium	Employ more sophisticated steganography algorithms that are more resistant to detection.
Visual cryptography imperfections	The decoded image may have imperfections due to the visual cryptography process.	Medium	Low	Use high-quality image processing techniques to minimize imperfections.

Table 4.1 Risk Analysis and Mitigation

CHAPTER-5

TESTING

5.1 TESTING PLAN

This testing plan outlines the various testing methodologies, test cases, and error handling strategies employed to validate the application's functionality and resilience.

Testing Methodologies:

Unit Testing: Individual modules and functions are tested in isolation to ensure their correct operation and adherence to specifications.

Integration Testing: Different modules are integrated and tested as a cohesive unit to verify their interactions and data flow.

System Testing: The entire web application is tested as a complete system to validate its functionality, performance, and security against real-world scenarios.

Robustness Testing: The application is subjected to unexpected inputs, stress conditions, and edge cases to assess its resilience and ability to handle unexpected situations.

5.2 COMPONENT DECOMPOSITION AND TYPE OF TESTING REQUIRED

Component	Type of Testing
Image Upload	Functional, Input Validation, Error Correction
LSB Steganography Encoding	Functional, Error Correction
Visual Cryptography Encoding	Functional, Error Correction
Image Overlapping and Decoding	Functional, Visual Quality, Error Correction
LSB Steganography Decoding	Functional, Data Recovery
n share division	Functional, Visual Quality, Boundary value check

User Interface	Usability, Accessibility, Responsiveness
----------------	--

Table: 5.1

Category 1: Test Cases for Different Image Formats

Objective: Verify that the application can successfully encode and decode messages in images of different formats, including PNG, JPG, and JPEG.

Test Cases:

PNG: Encode a message in a PNG image and decode it successfully.






IMAGE	DIMENSIONS
	30 x 30
	50 x 50
	130 x 130

Table: 5.2

JPG: Encode a message in a JPG image and decode it successfully.

IMAGE	DIMENSIONS
	40 x 40
	60 x 60


	90 x 90
---	---------

Table: 5.3

JPEG: Encode a message in a JPEG image and decode it successfully.





IMAGE	DIMENSIONS
	20 x 20
	70 x 70
	90 x 90
	100 x 100

Table: 5.4

Category 2: Test Cases for Different Image Sizes

Objective: Verify that the application can handle images of varying sizes, ensuring that the encoding and decoding processes remain functional regardless of image dimensions.

Test Cases:

Encode a message in a small image, medium-sized image and large image and decode it successfully.


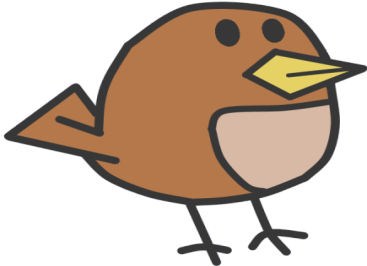

IMAGE	DIMENSIONS
	20 x 20
	512 x 512
	1024 x 1024

Table: 5.5

Category 3: Test Cases for Different Image Types (Colored and Black and White)

Objective: Verify that the application can successfully encode and decode messages in both colored and black and white images, ensuring that the color information does not interfere with the steganographic process.

Test Cases:

Encode a message in a colored image and black and white and decode it successfully.


IMAGE	PIXEL DETAILS
	Coloured pixels
<pre> 4,34,0,80,1,37,32,33,66,32,33,33,6 18,22,64,25,54,24,63,40,97,58,96, 34,88,68,90,62,78,4,98,17,3,84,17, 2,27,18,3,66,41,65,21,62,32,84,13, 51,89,43,46,30,18,84,61,93,28,73,6 7,73,93,23,0,38,15,37,54,22,52,77, 35,64,71,87,73,78,65,19,61,90,29, 98,29,69,47,37,39,85,81,11,14,90, 11,75,64,21,69,82,21,39,71,37,93,3 7,83,46,88,15,45,17,32,38,33,3,47, 25,68,20,10,23,78,2,53,50,54,27,50 61,63,9,86,100,40,20,44,73,100,27 36,97,13,86,43,14,68,28,27,62,55, 1 22 60 60 22 40 95 24 100 77 51 61 </pre>	Grayscale pixels

Table: 5.6

Additional Testing Considerations:

Message Length: Test the application's ability to handle messages of varying lengths, ensuring that the encoding and decoding processes remain efficient and accurate regardless of message size.

Image Quality: Evaluate the impact of image quality on the encoding and decoding processes, ensuring that message integrity is maintained even with compressed or low-quality images.

Error Handling: Test the application's error handling mechanisms, ensuring that it can gracefully handle unexpected scenarios and provide appropriate feedback to the user.

5.3 ERROR AND EXCEPTION HANDLING

Error Handling in main.py

The try-except block in main.py is used to handle the potential error of not being able to open the image file selected by the user. The user is prompted to upload an image file, and if the file is not uploaded or is not of a valid format, an error message is displayed. This error handling prevents the application from crashing if the user provides an invalid image file.

```

img = st.file_uploader('Upload image file', type=['jpg', 'png', 'jpeg'])

if img is
not
None:

    img = Image.open(img)

    try:

        img.save('images/img.jpg')

    except:

        img.save('images/img.png')

```

Error Handling in lsb_stego.py

The try-except block in lsb_stego.py is used to handle the potential error of not being able to open the image file that contains the encoded data. The lsb_decode function attempts to open the image file, and if the file is not found or is not of a valid format, an error message is returned. This error handling prevents the application from crashing if the user provides an invalid image file.

```

try:

    image = Image.open(file_name, 'r')

except:

    print('Error opening image file')

    return None

```

These try-except blocks are essential for ensuring that the application can handle unexpected errors gracefully and provide appropriate feedback to the user. Without these error handling mechanisms, the application could crash or behave unpredictably, leading to a poor user experience.

Input Validation Example (Encode Section):

```

if len(txt) == 0:

    st.warning('No data to hide')

elif img is None:

    st.warning('No image file selected')

```


The script checks if the user has provided both a message (txt) and an image (img) to encode. If either of them is missing, it displays a warning message to the user.

Boundary Checks Example (N-share Division Section):

```
no_of_shares = st.number_input('No. of shares',min_value=1, step=1)
```

The script uses `st.number_input` to allow the user to input the number of shares they want to generate. The `min_value=1` parameter ensures that the user cannot input a value less than 1, thereby enforcing a minimum boundary for the number of shares.

5.4 LIMITATIONS OF THE SOLUTION

Image Size Limitations:

The project web application is currently limited to handling images of a specific size range. This limitation arises from the computational complexity of both steganography and visual cryptography algorithms. As the size of an image increases, the computational cost of these algorithms grows exponentially, making it impractical to process large images in real time.

To address this limitation, future work could explore more efficient steganography and visual cryptography algorithms that can handle larger images without compromising performance or security. Additionally, implementing parallel processing techniques could help distribute the computational load and improve scalability.

Image Quality Degradation:

The steganography process involved in hiding secret messages within images may introduce some visual artifacts into the encoded image. These artifacts are caused by the subtle modifications made to the pixel values of the image to embed the message. While these artifacts may not be noticeable to the naked eye, they can become more apparent when the image is zoomed in or subjected to image processing techniques.

To minimize image quality degradation, future work could investigate alternative steganography methods that are less prone to introducing visual artifacts. Additionally, implementing adaptive steganography techniques that adjust the embedding strength based on the local image characteristics could help preserve image quality while maintaining message integrity.

Visual Cryptography Imperfections:

The decoded image obtained from visual cryptography may exhibit minor imperfections due to the inherent nature of the technique. Visual cryptography relies on the overlap of two or more image

shares to reveal the hidden message. However, due to the discrete nature of pixel values, the edges of the decoded image may appear slightly jagged or distorted.

To address these imperfections, future work could explore alternative visual cryptography techniques that utilize more sophisticated encoding and decoding algorithms.

Steganalysis Vulnerabilities:

The project web application may be susceptible to advanced steganalysis techniques. Steganalysis aims to detect the presence of hidden messages within digital media, such as images. While the specific steganography algorithm used in the application is designed to be resistant to common steganalysis methods, more sophisticated techniques may be able to detect the presence of hidden messages.

To enhance the security of the application, future work could investigate more robust steganography algorithms that are less susceptible to steganalysis. Additionally, implementing techniques such as stegano-noise and cover image selection could further improve the resistance against steganalysis attacks.

Security of Encryption Techniques:

The security of the RSA and Caesar Cipher encryption techniques used in the project may be susceptible to vulnerabilities if not implemented with strong key management practices and proper key lengths. Future work could focus on enhancing encryption security by adopting more robust cryptographic algorithms and key management strategies.

Message Length and Capacity:

The capacity to embed messages within images using steganography may be limited by the size of the image and the strength of the steganographic technique. Larger messages or messages requiring higher security might exceed the capacity of the chosen steganography approach. Exploring advanced steganographic methods or combining multiple techniques could improve message capacity and security.

Performance Impact during Image Processing:

The computational overhead introduced by steganography, encryption, and visual cryptography algorithms may impact the overall performance of the web application, particularly when handling multiple concurrent requests or large volumes of data. Optimizing algorithms, implementing caching mechanisms, and leveraging hardware acceleration (e.g., GPU processing) could mitigate performance bottlenecks.

CHAPTER-6

FINDINGS, CONCLUSION AND FUTURE WORK

6.1 FINDINGS

1. **Security Enhancement:** The combination of both symmetric (Caesar Cipher) and asymmetric (RSA) encryption techniques provides a robust security framework for transmitting confidential images. This dual-layered encryption approach significantly increases the complexity of decryption for unauthorized parties.
2. **Share Distribution Efficiency:** The use of secret sharing algorithms ensures efficient distribution of encrypted image shares among shareholders. This approach enhances data confidentiality by requiring collaboration among shareholders to reconstruct the original image.
3. **Decryption Integrity:** The decryption process, involving RSA decryption followed by Caesar Cipher decryption, ensures the integrity of the reconstructed image. Verification of minimum share requirements and correct key usage further enhances the security and reliability of the decryption phase.
4. **User Access Control:** The system's design includes mechanisms for user access control, allowing only authorized shareholders who have agreed to merge their shares to access the decrypted image. This controlled access mechanism adds an extra layer of security to sensitive information.
5. **Ease of Use:** The proposed algorithm offers a user-friendly approach, with clear steps for both encoding and decoding processes. The intuitive design facilitates seamless encryption, distribution, reconstruction, and decryption of secret images, making it accessible to users with varying levels of technical expertise.
6. **Overall Security Framework:** Collectively, the findings suggest that the proposed algorithm provides a comprehensive and effective security framework for transmitting confidential messages. By combining multiple encryption techniques with secure distribution and controlled access mechanisms, the algorithm ensures the confidentiality, integrity, and reliability of sensitive information transmission.

6.2 CONCLUSION

Sharing data secretly in a collaborative manner is very important. The proposed solution not only makes it difficult for intruders to steal the data but also makes it nearly impossible as we encrypt the data before dividing into shares. Encryption provides extra security for the data in addition to that data being divided into shares. Now even if the intruder has a required number of shares, he won't be able to get the original image as it is as he doesn't know the values of the key. What makes

this algorithm good is that the intruder has to get all the required number of shares in the first place and this is tough. What is tougher is that even if he has the required number of shares, he should have the key values as well. So, it's better if the value of 'k' is closer to the value of 'n' and also has a good set of keys for encryption.

6.3 FUTURE WORK

This proposed method has a compulsion that there should be only one admin and each shareholder has to take the shares from and has to bring back the shares to that only admin. It is that person who initially takes the image to be shared secretly, encrypts the image, divides into 'n' shares, sends mails, combines the accepted shares, decrypts the image, gets back the secret image and maintains all the files related to that image. It can be further extended in such a way that encryption, division into 'n' shares and mailing those shares be done at one end taken care by one admin and combining 'k' shares and decryption be done at another end taken care by another admin. This makes this algorithm more feasible and robust to use and work with. Also, with the changing technologies, newer and stronger encryption algorithms can replace the ones that are used in this proposed Algorithm.

This same idea can be used in other fields as well like in providing authentication in Photography Contests, Online Voting Systems using Visual Cryptography, in Copyright authentication etc.

REFERENCES

- [1] A. Shamir, "How to Share a Secret," Communications of the ACM, vol. 22, no. 11, Nov. 1979.
- [2] J. Sharmila and J. Gurralla, "A Novel Approach on Secure Data Transfer for General Transactions using Secret Sharing Scheme," International Journal of Computer Applications, vol. 172, no. 8, Aug. 2017.
- [3] S. Kandar and A. Maiti, "K-N Secret Sharing Visual Cryptography Scheme for Color Image Using Random Number," International Journal of Engineering Science and Technology (IJEST).
- [4] M. Naor and A. Shamir, "Visual Cryptography" (The preliminary version of this paper appeared in Eurocrypt 94).
- [5] K. Shankar and P. Eswaran, "A new k out of n secret image sharing scheme in visual cryptography," Jan. 2016, doi: <https://doi.org/10.1109/isco.2016.7726969>.
- [6] S. Bhuyan, "Image Security using Visual Cryptography," Thesis, NIT Rourkela.
- [7] D. Poddar, "(2, N) Visual Cryptographic Scheme for Black and White Pixels Using Square Matrices," Indian Statistical Institute, Kolkata, July 2016.
- [8] K. Shankar and P. Eswaran, "Sharing a Secret Image with Encapsulated Shares in Visual Cryptography," in 4th International Conference on Eco-friendly Computing and Communication Systems, ICECCS 2015.
- [9] M. Naor and A. Shamir, "Visual cryptography," in Advances in Cryptology-Eurocrypt'94, 1995, pp. 1-12.
- [10] J. Weir and W. Yan, "Visual Cryptography and Its Applications."