PROJECT REPORT

SBS CS 01 03 20 C 0042

ON

"Crypagno graphy-a innovative combination of cypto with steganography"

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARDOF THE DEGREE OF

MASTER OF COMPUTER APPLICATION

Session(2022-2024)

Submitted By:

Dheeraj Kumar
Roll No:220563

Neeraj Kumar
Roll no. 220592

Submitted To:

Dr. Suraj Arya

Assistant Professor

Central University of Haryana
Department of Computer Science & Information Technology

Student's Declaration

We here by declare that the project entitled"CRYP-AGNO GRAPHY" submitted in the partial fulfillment of requirement for award of the degree of Master of Computer Application to Central University of Haryana, is my authentic work carried out during the 3rd semester of MCA under the supervision of **Dr. Suraj Arya.** This project has not formed the basis for

award of any other degree, associate ship, fellowship or any other similar title.

Dheeraj kumar
Roll No.:-220563
Session:-2022-2024

Neeraj Kumar
Roll No.:-220592
Session:-2022-2024

Central University of Haryana

Department of Computer Science and Information Technology

This is to certify that this report entitled "CRYP-AGNO GRAPHY" is a bonafied record of the project presented by Dheeraj Kumar(220563), Neeraj Kumar(220592).Session 2022-2024 a student of Master of Computer Application at Department of Computer Science and Information Technology, Central University of Haryana. He/ She has completed the project work under my supervision.

Date: …/…/…

Dr Suraj Arya

Assistant Professor

Computer Science and Information Technology ,Central University of Haryana

Central University of Haryana

Department of Computer Science and Information Technology

This is to certify that this report entitled "CRYP-AGNO GRAPHY" is a bonafied record of the project presented by Dheeraj Kumar(220563), Neeraj Kumar(220592).Session 2022-2024 a student of Master of Computer Application at Department of Computer Science and Information Technology, Central University of Haryana. He/ She has completed the project work under my supervision.

Date: …/…/…

Dr Keshav Singh Rawat
Associate Professor
Computer Science and Information Technology ,Central University of Haryana

ACKNOWLEDGEMENT

Presentation inspiration and motivation always played a key role in the success of any venue. First of all, We would like to thank the Almighty God who is the one who has always guided me to work on the right path of life. Without his grace the project could not become the reality.

We would like to express our special thanks of gratitude to our project guide and supervision, **Dr.Suraj Arya**, Assistant Professor, Department of Computer Science and Information Technology, for his valuable suggestions and advices.

We would also like to express our gratitude to **Dr. Keshav Singh Rawat**, Associate professor, Head of Department of Computer Science and Information.

We would also like to extend special thanks to the entire staff for their full cooperation, guidance and support.

We would like to thanks our friends who helped us so much to completion of this project. Last but not the least, we would like to thank our parents, for the support and affection they had given to us throughout this course and for always encouraging our to complete my work.

Dheeraj Kumar
Roll No.:-220563
Session:-2022-2024

Neeraj Kumar
Roll No.:-220592
Session:-2022-2024

Content

Front Page	1
Student's Declaration	2
Certificate 1	3
Certificate 2	4
Acknowledgement	5
Content	6
Figure Index	7
Table Index	8
1.Introduction and Objective of the Project	
1.1 Introduction	9
1.2 Aims and Objective	10
2.System Specifications	
2.1 System specification	12
3. Problem definition and requirement	
3.1 Problem Definition	13
3.2 Requirement Specification	14
4.Analysis	16
5. System Design, Coding and Testing	
5.1 System Design	18
5.2 Algorithms Used	20
5.3 Testing	24
6. Result	25
7. Future Scope	31
8. Conclusion and Bibliography	
8.1 Conclusion	32
8.2 Bibliography	33
9. Annexure	34

Table

Table: 1 Hardware Requirements	12
Table: 2 Software Requirements	12

Figure index

Figure No.	Page No.
Figure 1	16
Figure 2	18
Figure 3	19
Figure 4	25
Figure 5	25
Figure 6	26
Figure 7	26
Figure 8	27
Figure 9	27
Figure 10	28
Figure 11	28
Figure 12	29
Figure 13	29
Figure 14	30
Figure 15	30

Chapter -1: Introduction and Objective of the Project

1.1 Introduction

Two techniques, Cryptography and Steganography is often used for assurance classification, integrity and accessibility of data. Combination of cryptographic methods and steganography provides a robust and secure system communication system for mutual sharing information over an insecure channel. code is a technique for maintaining the confidentiality of a texting alone. It will protect confidential information by encoding it in an indecipherable format. It is a protection and exchange mechanism Use secret code. Cryptography defines security communicate in full view of vengeful strangers considered enemies. Encryption performs calculations and a secret key to edit contributions to code. A certain mathematical assessment will be given convert plain text to cipher text if the secret is similar key is used.

Cryptography valuations remain secure if the attacker cannot complete any plaintext or characteristics of the secret key. These calculations are of two types symmetric key and asymmetric key. Symmetric lock cryptography is a type of encryption in it a key, i.e. a secret key, is used for encryption and decode electronic data. This encryption method in contrast to asymmetric key cryptography, in which two or three public keys and a private key are used to encode and decode messages.

Steganography is the practice of concealing messages, data, or information within text, images, or other non-confidential data to conceal the existence of the hidden information. Unlike cryptography, which aims to make the content of a message unreadable, steganography aims to keep the fact that communication is taking place private.

Below are some key concepts and techniques related to steganography:

Carrier: This is the file or data that will contain hidden information. Common media include images (JPEG, PNG, BMP), audio files (MP3, WAV), videos (AVI, MP4), and even text.

Payload: This is hidden information that must be transmitted or stored secretly. This can be text, files, images or any other data that needs to be hidden.

Embedding: The process of hiding the load in the carrying medium is called embedding. This involves manipulating the medium so that the changes are subtle and do not attract attention.

Key: Some steganographic techniques require a key or password to embed or extract hidden information. This adds an extra layer of security and process control.

Detection and Extraction: To recover hidden information, one must use appropriate techniques to detect and extract the payload from the medium. This process often requires a specific key or algorithm.

Steganography can be classified into different types depending on the medium used:

Image steganography: Hiding data in digital images.

Audio steganography: Hide information in audio files.

Video steganography: embed data into video files.

Text steganography: Hiding information in text.

Steganography is often used in conjunction with encryption to improve communication security. While encryption protects the content of the message, steganography helps hide the fact that communication is taking place. It is important to note that although steganography can be used for legitimate purposes, such as protecting sensitive information, it can also be used for malicious purposes, including including confidential communications for illegal purposes. As a result, there are legal and ethical considerations regarding the use of steganography.

1.2 Aims and Objective

The main goal of steganography is to embed information in a way that is difficult to detect or decode. This can be achieved by subtly changing the look and feel of the media, be it images, audio files, videos or even text without raising suspicion. The medium is often called the "envelope" or "container" and the hidden information is called the "payload" or "message.

Goals of Steganography :

Develop a secure communication tool: To provide a platform for users to confidentially exchange information while ensuring the confidentiality of communications.

Improved privacy and data security: Improve data security and confidentiality through the implementation of steganographic methods. Allows users to protect sensitive data by hiding it in innocuous-looking media, preventing unauthorized access.

Educational and research goals: To contribute to the understanding and improvement of steganography concepts and techniques. To facilitate research and education in the field of information security and cryptography.

Integrates with encryption for enhanced security: Integrate steganography with encryption algorithms for a stronger security solution. Combines the strengths of encryption and steganography to ensure both the security and confidentiality of communications.

Objectives:

Develop user-friendly interface: Design an intuitive and user-friendly interface for the steganography tool.

Perform steganographic techniques: Implement steganographic techniques, such as Least Significant Bit (LSB) embedding, frequency domain methods, and others. Flexibility in information concealment across different types of media.

Guaranteed solidity and imperceptibility: Ensure that integrated information is resistant to common attacks and is imperceptible to the human senses. Resistant to cryptographic techniques, maintaining media quality.

Integrated encryption algorithms: Integrate encryption algorithms to secure hidden information. Improved security through a combination of steganography and encryption.

Performance optimization: Optimize the performance of the steganography tool in terms of speed and resource usage. Efficient data processing with minimal impact on system resources.

Documents and instructions for use: To provide comprehensive documentation and user guides for the steganography tool. Clear instructions, troubleshooting guides, and helpful resources for users.

Test and evaluate: Conduct thorough testing to evaluate the functionality and security of the tool. Identify and remediate all vulnerabilities, ensuring the tool meets security standards. By defining clear goals and objectives.

Chapter -2: System Specifications

2.1 System Specifications:

Required hardware components and software requirements needed for effective andefficient running of the Software:-

Table 1: Hardware Requirements

S.No.	Hardware Required	Specification
1.	Processor Speed	2.4 GHZ Processor
2.	RAM	4 GB RAM
3.	Memory Storage	500 GB

Required Software components and software requirements needed for effective andefficient running of the Software:-

Table 2: Software Requirements

S.No.	**Software**	**Minimum System Requirement**
1.	Operating System	Windows 7,10,11 or MAC OS 10.8,10.9, or10.11, LINUX
2.	Runtime Environment	Visual Studio code, Pycharm
3.	Tools and Technology	Python

Chapter -3: Problem Definition and Requirement Specifications

3.1 Problem Definition

In a rapidly evolving digital landscape, creating and deploying image steganography technology has become imperative due to the growing need for secure and confidential communications. Traditional methods of ensuring data privacy, such as encryption, while effective, may not adequately address the need to exchange confidential information. This necessity arises from the inherent limitations of encryption, which is primarily intended to make the content of a message unreadable without fully concealing the existence of the communication. Image hiding technology provides a solution to this limitation, providing a way to hide sensitive information in digital images without arousing suspicion. The need for visual steganography is underscored by persistent vulnerabilities in existing secure communications tools as well as continued advances in digital forensics and cryptanalysis techniques. . By embedding secret data in seemingly harmless images, steganography not only improves communications security but also adds a layer of obscurity, making it difficult for adversaries to detect or intercept communications. secret. Image steganography is therefore essential for individuals and organizations looking for a comprehensive approach to protecting their sensitive information, ensuring both privacy and decision-making in digital field.

However, current steganography tools may have vulnerabilities, making them vulnerable to detection and analysis. Additionally, there is a need for a user-friendly and accessible image encryption tool that seamlessly integrates with encryption methods to create a robust and comprehensive security solution. This project aims to address these challenges by developing an advanced visual steganography system that ensures not only the transmission of confidential information but also resilience to potential attacks and easy to use for a broad user base.

3.2 Requirement Specifications

The requirements specification for an image cryptography system describes the functional and non-functional requirements as described below:-

Functional Requirements:

a) User Interface: The steganography tool must have an intuitive and user-friendly interface that is easy to use by users with efficient technical expertise.
b) Embedding Algorithms: Implements multiple steganographic embedding algorithms, such as Least Significant Bit (LSB) embedding algorithm.
c) Encryption Integration: Adding encryption algorithms to improve the security of hidden information and provide an additional layer of protection.
d) Payload Specification: Allows users to specify the type and format of the payload to embed (text, file, image) and ensures compatibility with different data types.
e) Detection and Extraction: Provides a mechanism to detect and extract hidden information from media using appropriate encryption keys and ciphers.
f) Transportation support: Supports multiple media, including popular image formats (JPEG, PNG,BMP), providing user flexibility. Detect and extract:
g) Robustness and Imperceptibility: Ensure that embedded information is resistant to common attacks and imperceptible to the human senses, while maintaining media quality
h) Key Management: Includes a key management system for cryptographic and encryption keys, allowing users to securely create, store, and manage their keys.
i) Documentation and help system: Provides comprehensive documentation, user guides, and in-app contextual help to help users understand and use the tool effectively.

Non-Functional Requirements:

a) Protection: The system must meet high security standards, ensuring the confidentiality and integrity of integrated information.
b) Usability: Steganography tools should be user-friendly, have a short learning curve, and be intuitive to navigate, catering to users with different technical backgrounds.
c) Efficiency: The system must operate efficiently, minimizing processing time to integrate and extract information.
d) Scalability: The tool must be scalable to handle different data sizes and accommodate future enhancements.
e) Reliability: Ensure system reliability by minimizing errors, handling difficult cases, and providing error recovery mechanisms.
f) Compatibility: The steganography tool must be compatible with major operating systems (Windows, Linux, macOS) and support a variety of image editing software.
g) Maintainability: System design with modular components and clear documentation for easy maintenance and updates.

Chapter -4: Analysis

The Below fig. shows the all service categories and their sub-categories provided by our purposed project.

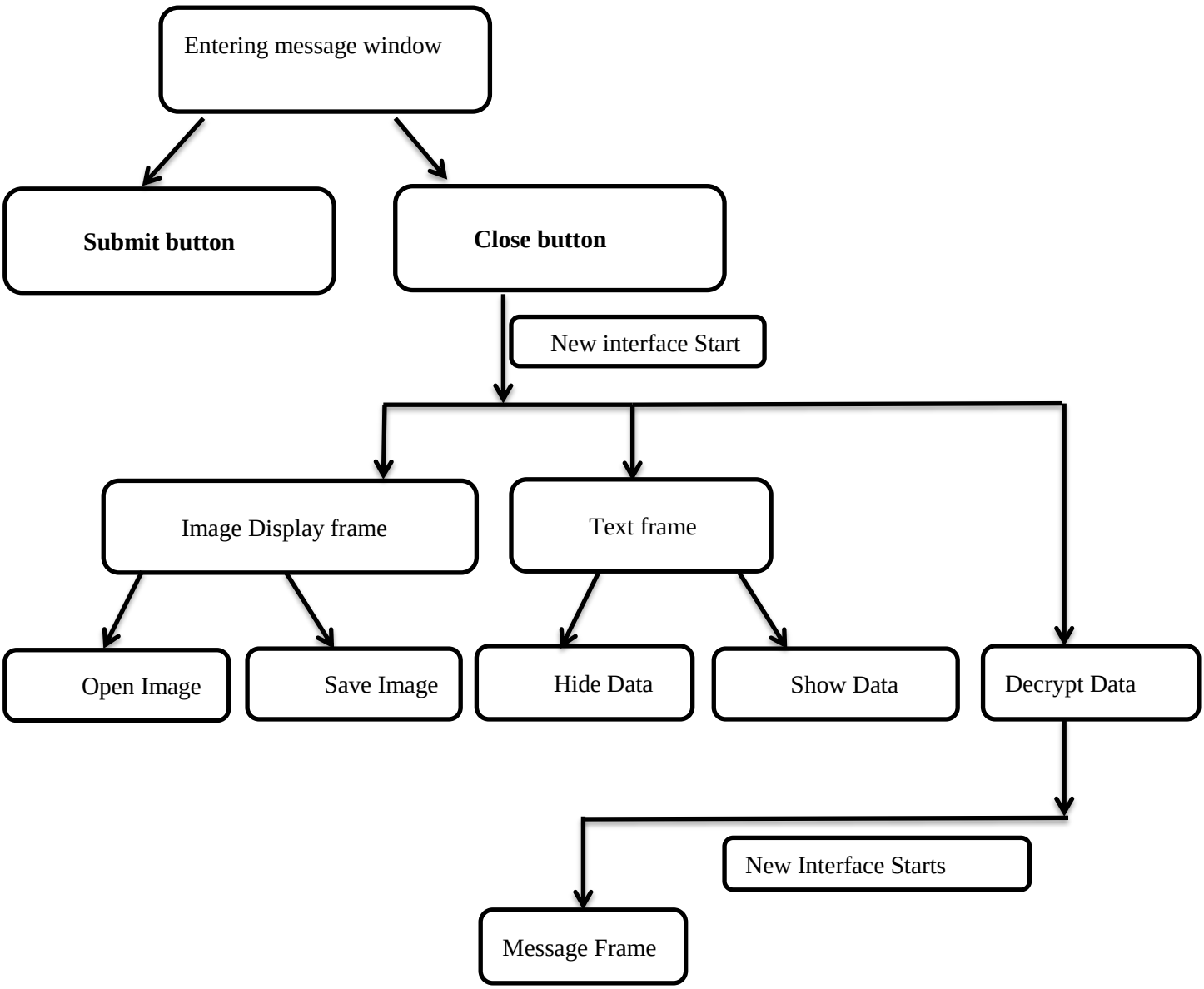


Figure: 1

Our purposed system contains following tasks:-

a) Enter message Window ;- Here user enters the message thet is confidential.

b) Submit Button:- It submits the data entered by user.

c) Close Button:- It close the current interface and redirects the user to new interface Where user can perform different tasks.

d) Image Display Frame:- It display the image selected by the user which provides two option to the user as shown below:
   1) Open Image:- This button opens the window through which user can select an image.
   2) Save Image:- This button saves the stego image in which data is hidden.

e) Text Frame:- It shows the encrypted text of the message entered.

f) Hide Data:- It will hide the encrypted text in the image selected by the user using LSB technique.

g) Show Data:- It will show the data which will be hidden in the stego image which will be received from the other user.

h) Decrypt Data:- It will decrypt the data which is extracted from the stego image received from the other user and redirects to the new interface which prints the confidential message sent by the other end user.

Chapter -5: System Design , Coding And Testing

5.1 System Design

The below flow chart shows the design of our purposed system. This figure shows the flow of the system for a user:

Figure 2: Process of Data Hiding:-

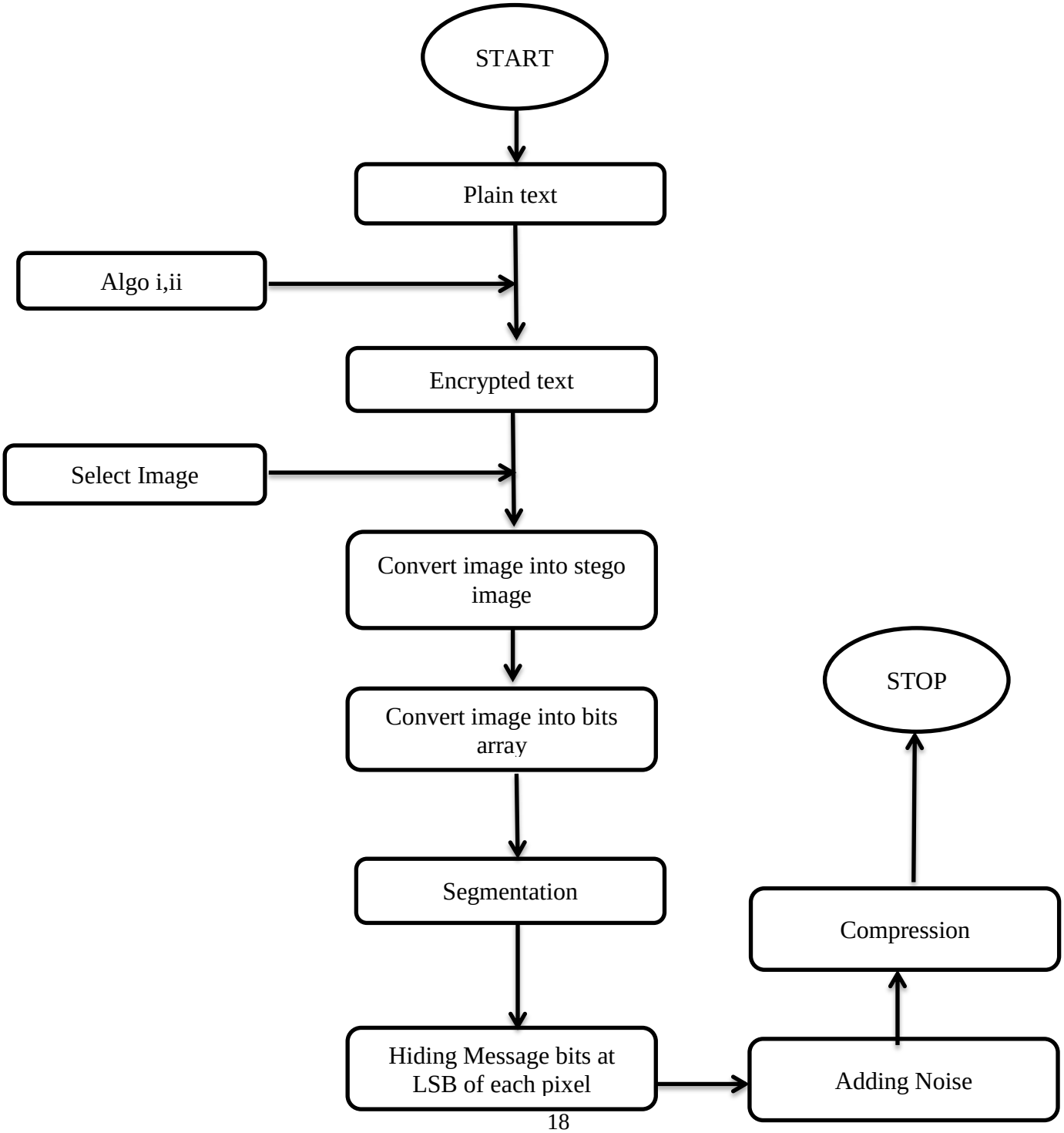
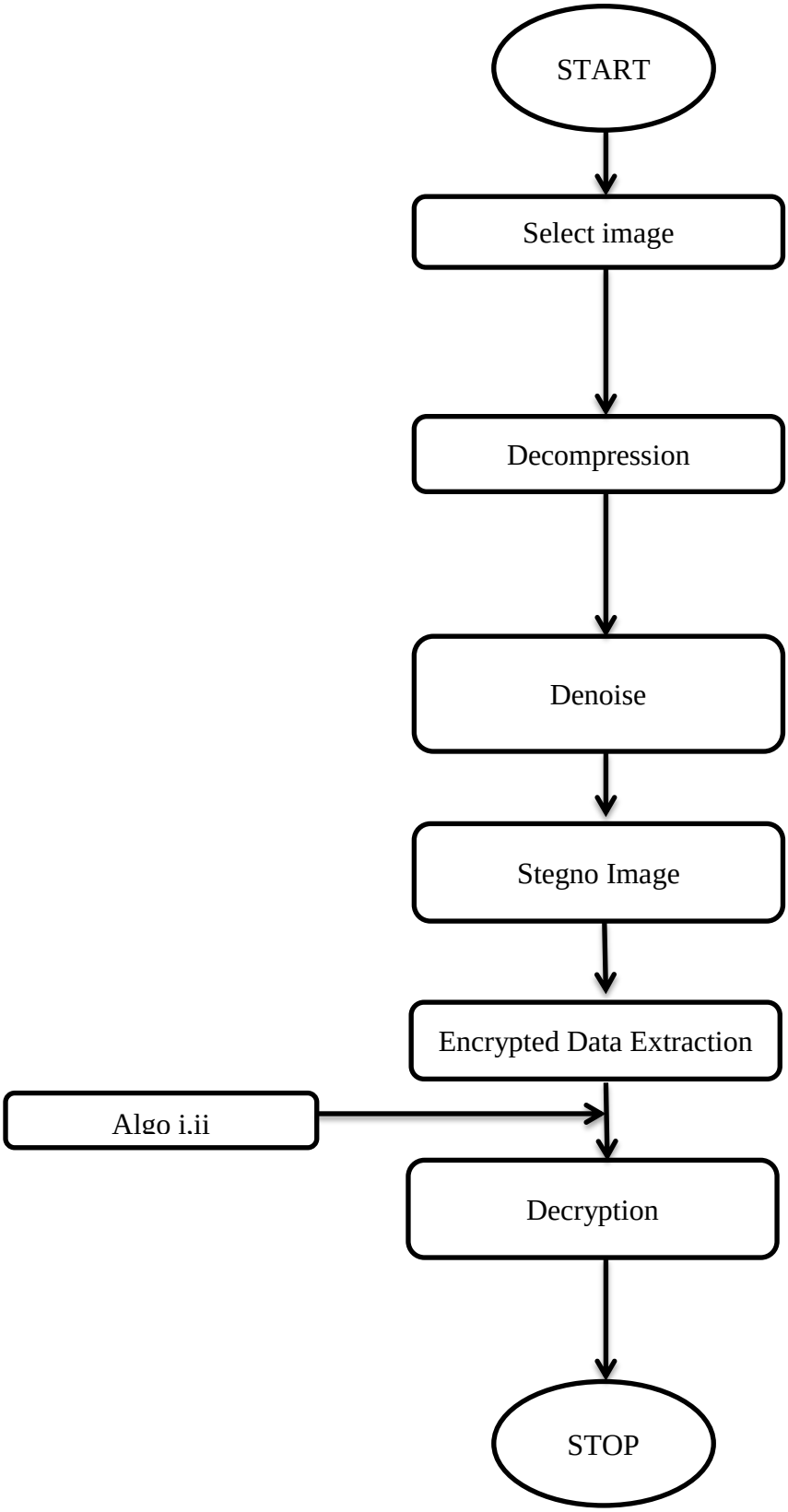


Figure 3: Process of Data Extraction :-



5.2 Algorithms Used:

Encryption 1 :-

1) make a dict string varaible which contain all symblos and letters
2) enter msg and key
3) msg and key will be sent to encrypt function as argument
4) take a empty encmsg msg
5) for i in msg

   loc=key+dict.index(i) #loc is variable to store location of each crctr in each turn

   loc=loc%len(dict) #loc is divided by lenth of dict so that location never biger than dict lenth

   encmsg+=dict[loc] #each newly text is append in empty strng encmsg

6) encrypted msg is returned as encmsg

Decryption 1:-

1) key is enterd
2) encrpt msg and key is passed to decrpt function as argumnt
3) take a empty decrpt msg
4) for i in encmsg

   loc=dict.index(i)-key #first finds the encrpt crctr location then key value is subtractd and finds orignal crctr

   loc=(loc+len(dict))%len(dict)

   decmsg+=i #orgnl crctr is placed in emty decmsg

5) return decmsg

Encryption 2:-

1) make a dict string varaible which contain all symblos and letters
2) enter msg
3) random function is executed for creation of key and it stored in ".txt"file
4) this key file is being read and msg and key is passed to encrypt function as argument
5) take a empty encmsg msg
6) two chrctr array are creatd one stores the all dict symbols and other contain the key symbls
7) for i in msg:
   index=charA.find(i) #here finds the index of each crctr in msg from crctr in char arrayA which contain dict symblos
   encmsg+=charB[index] #append each crctr at index in charB which contain key symbls
8) return encmsg

Decryption 2:-

1) here aftr reads the ".txt" file key and encrpt msg is passed to decrpt function
2) make a empty decmsg
3) previous two chars array are taken
4) charsA,charsB=charsB,charsA
5) for i in x:
   index = charsA.find(i) #here finds the index of each crctr in msg from crctr in char arrayA which contain dict symblos
   dcmsg += charsB[index] #append each crctr at index in charB which contain key symbls
6) return decmsg

Technologies used in our project:-

1) **Noise:-** In digital imaging, "noise" refers to unwanted variations in pixel values that can distort the visual quality of an image. There are many different types of noise, each coming from different sources and affecting images differently. Common types of image noise include Gaussian noise, which appears as random variations in intensity; salt-and-pepper noise, characterized by isolated bright and dark pixels; and speckled noise, which appears as granular patterns. Other forms of noise include quantization noise, transient noise in video footage, and color noise that affects color channels. Noise reduction techniques include the use of filters, denoising algorithms, and image enhancement methods to reduce the effects of noise and improve overall image quality. Noise recognition and processing is critical in fields such as image processing, photography, and medical imaging to ensure accurate interpretation and analysis of visual information.

2) **Compression:-** Image compression is a process of reducing the size of a digital image file while maintaining an acceptable level of image quality. The main goal is to minimize the amount of data needed to represent an image, thus making storage and transmission more efficient. Compression is important to save storage space, reduce transmission bandwidth, and improve overall system performance.

Related work:-

A. TRADITIONAL BASED STEGANOGRAPHY METHODS:-

Usually, the least significant bit (LSB) substitution method is used to perform steganography. Images usually have higher pixel quality, not all pixels are used. LSB methods operate under the assumption that changing a few pixel values will show no visible change. The secret information is converted into binary form. The cover image is scanned to determine the least significant bits in the noise region. The binary bits of the secret image are then substituted into the LSB of the cover image. The alternative method must be implemented with caution because loading too many cover images can result in display changes that reveal the presence of confidential information. With the LSB method as a reference, several related methods have been proposed. For example, a slight variation in converting the secret message to binary code. Huffman encryption method is used to encode the secret

message into binary bits. The encoded bits are then integrated into the cover image using the LSB method. Another version of the LSB method is used for RGB images. The cover art has 3 channels and is cut into pieces. The secret message is integrated in all three planes in a ratio of 2:2:4 for the R, G and B planes. The frequency domain is exploited in the quantum imaging domain, and the pixels are considered to influence the colors used to hide the secret bits. A combination of cryptography and steganography is used where the LSB of the cover image is replaced by the most significant bits of the secret image. A pseudorandom number generator is used to select the pixels and the key is encrypted by rotating each time. The k-LSB method is proposed in which the least k bits are replaced by the secret message.

B. Latest methods used nowadays:-

➔ **CNN-BASED STEGANOGRAPHY METHODS:-**

Imagedecoding using CNN models is mainly based on encoder-decoder architecture. Two inputs: the cover image and the hidden image are given to the encoder to generate and output the stego image and the stego image is given as input to the decoder to generate the hidden image. The basic principles are the same but for the different methods we tested on different architectures. . The method of combining the input mask image and the hidden image is a different and unfocused approach, but the differences between convolutional layers and layers are relevant. The number of filters used, steps, filter size, activation function, and loss function used differ between the methods. The important thing here is that the cover image and the hidden image must be the same size, so all the pixels of the hidden image will be distributed in the cover image.

➔ **GAN-BASED STEGANOGRAPHY METHODS**

GAN is a type of deep CNN introduced by Goodfellowet al. GAN uses game theory to train a generation-generation model for image processing. Two networks (generating network and classifier network) compete with each other to generate an optimal image in the GAN architecture. A generator model provides data and an output that closely approximates the given input form. A discriminative network classifies the generated images as fake or real. Both networks were trained to fit the model to the input data while minimizing noise. The discriminator model is trained to detect false images. Many different GANs have been proposed since then, making them more powerful and suitable for image processing tasks.

5.3 Testing

An summary of the extensive testing carried out on the steganography GUI platform is given in this testing report. Aspects such as functionality, security, user experience, and performance were all tested. The intention was to guarantee that the platform satisfies quality requirements and offers a smooth user and service provider experience.

i. Functional Testing:

    1. Conducted end-to-end testing of key features, including text encryption and data hiding.
    2. Verified the accuracy of compression and extracting data from image which is hide at LSB.

ii. Security Testing:

    1. Assessed the platform's security measures, including data encryption, secureauthentication, and protection against common vulnerabilities.
    2. Conducted penetration testing to identify and address potential security risks.

iii. Performance Testing:

    1. Simulated various usage scenarios to assess the platform's performance undernormal and peak loads.
    2. Measured response times, system resource usage, and identified any performancebottlenecks.

iv. Compatibility Testing:

    1. Tested the platform on different devices and platforms to ensure compatibility.
    2. Ensured that the platform is responsive and functional across various screen sizes and resolutions.

v. Regression Testing:

    1. Ensured that changes did not negatively impact existing functionality.

Chapter -6: Results

6.1 Interfaces:-

The figure below shows the First interface of our tool. Whenever any user will visit our tool this page will appear first. There are choices: He can write text which he wants to hide in image and also have to give key which is used for encryption decryption.

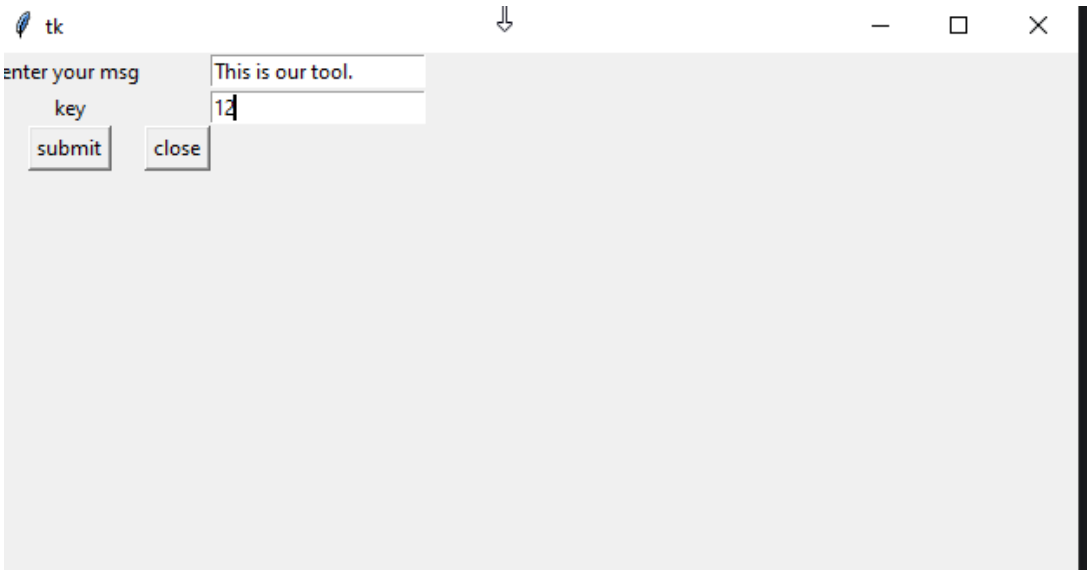


Figure:4

➔ Next we click on “submit” button and then click on “close” button and a new interface gets open on which encrypted text is shown and then we click on “open image” and select an ‘jpg image’ in which we have to hide data.

Figure:5

➔ Next we click on “Hide Data” button and then click on “save image” an compressed image with noise will be saved on the project directory as shown below:

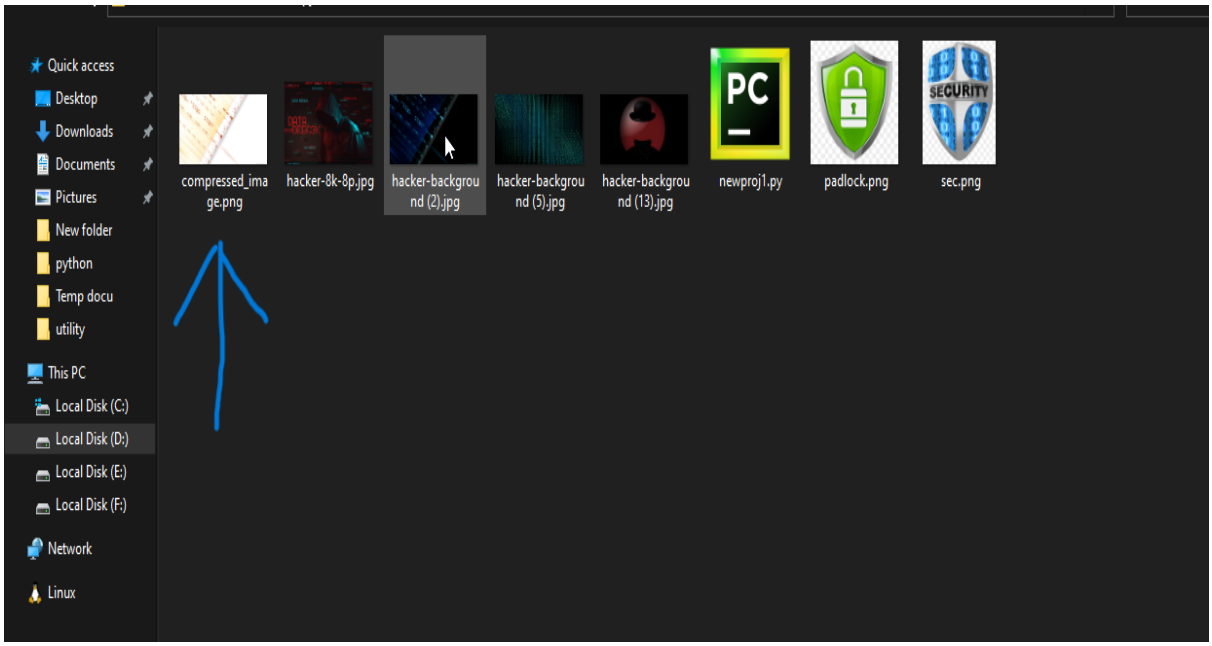


Figure:6

➔ Next we close the opened interface and again run the program and again enter the key which we used earlier and click on close as shown :

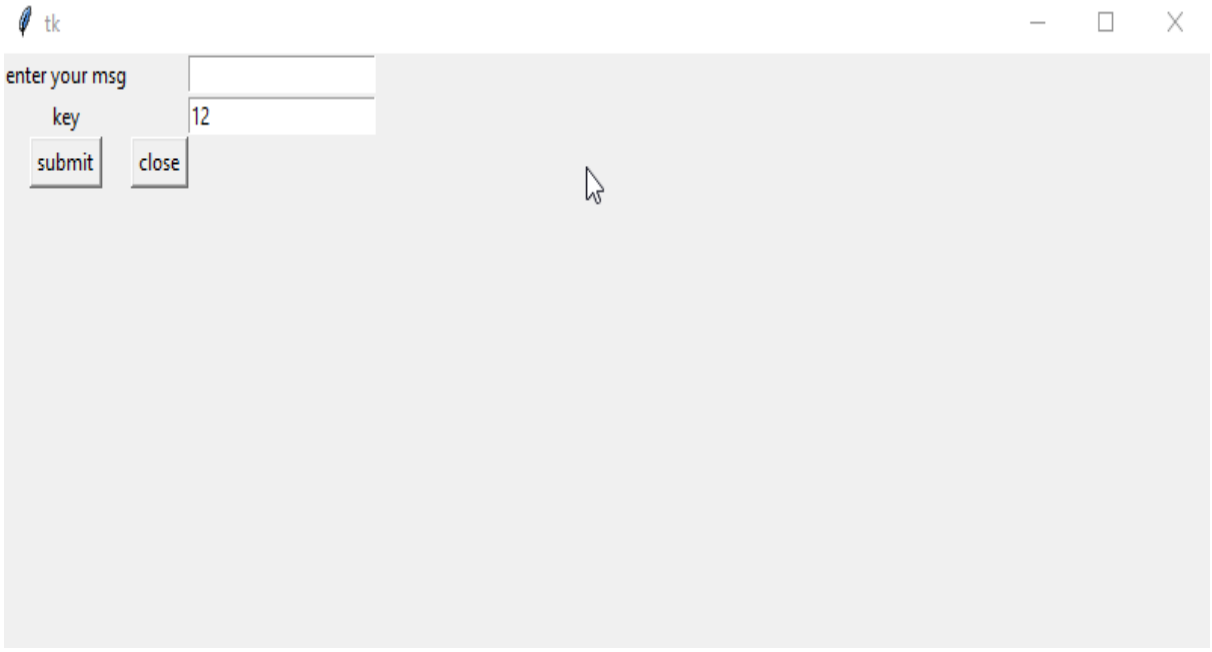


Figure:7

➔ As soon we click “close” button new interface opens again and we open image in which we hide data i.e, “compressed_image.png”. and then click on show data so that encrypted text appear in text frame as shown:

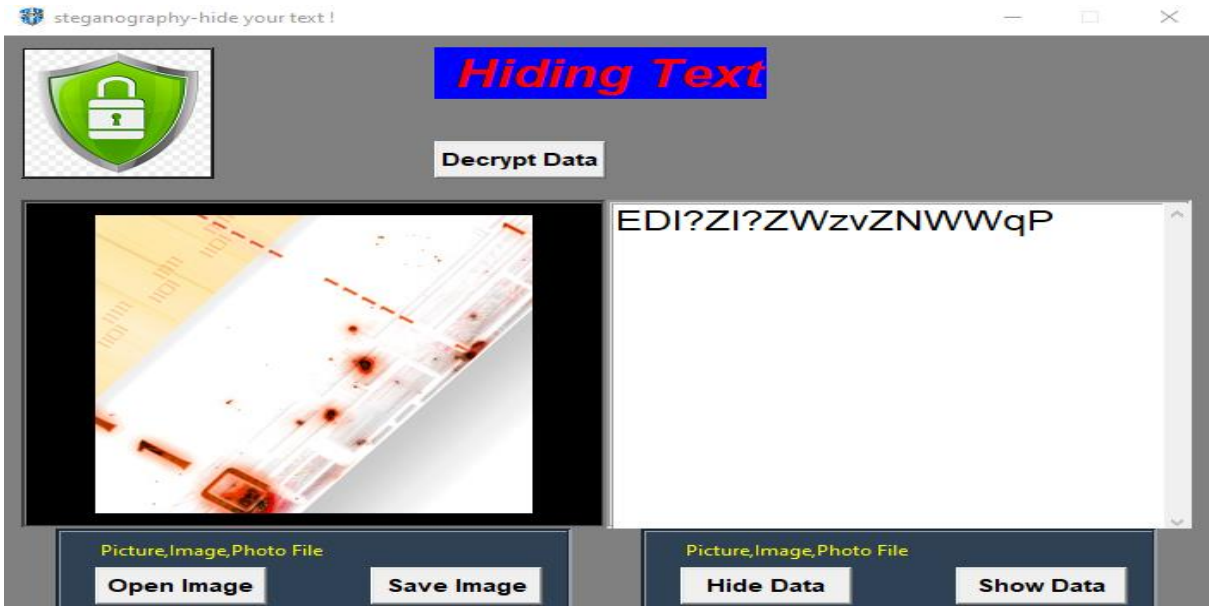


Figure:8

➔ Then click on Decrypt Data so that new interface gets open which shows the encrypted and decrypted text both which is our actual data as shown:

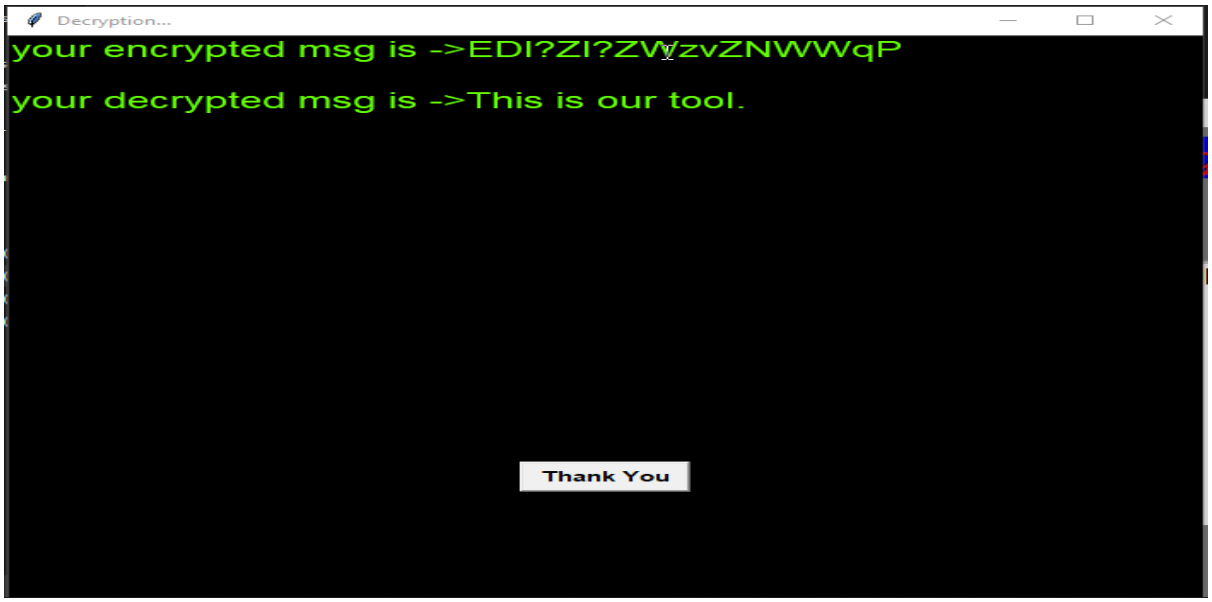


Figure:9

➔ If we use 0 in place of key field then our 2nd algorithm for encryption and decryption will be executed automatically and a “key2.txt” key file will be generated in project directory which is used for encryption and decryption both. Sender must have to send this files to the receiver with “compressed_image.png” As shown below:

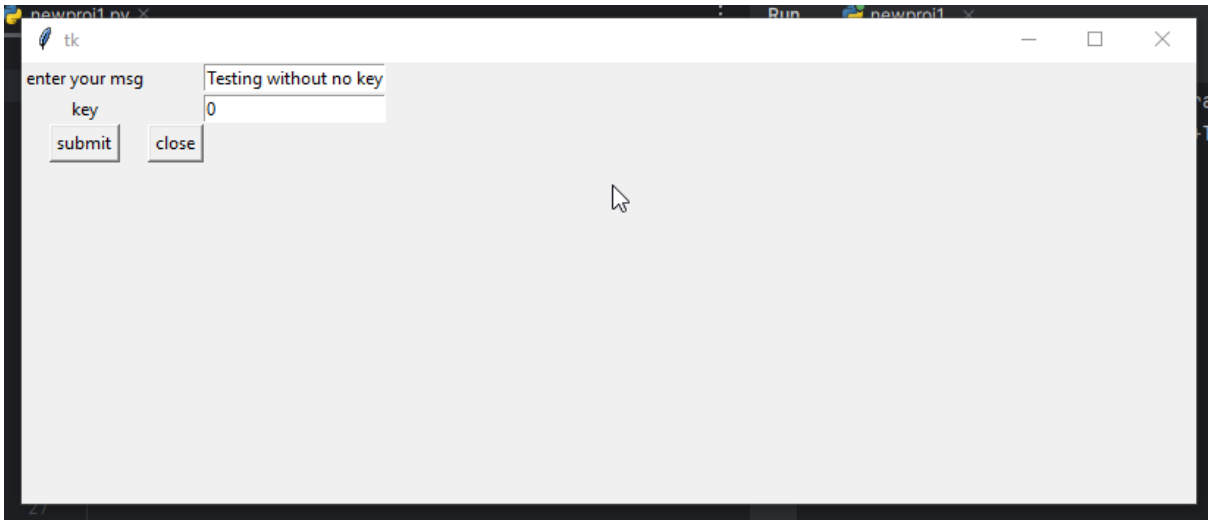


Figure:10

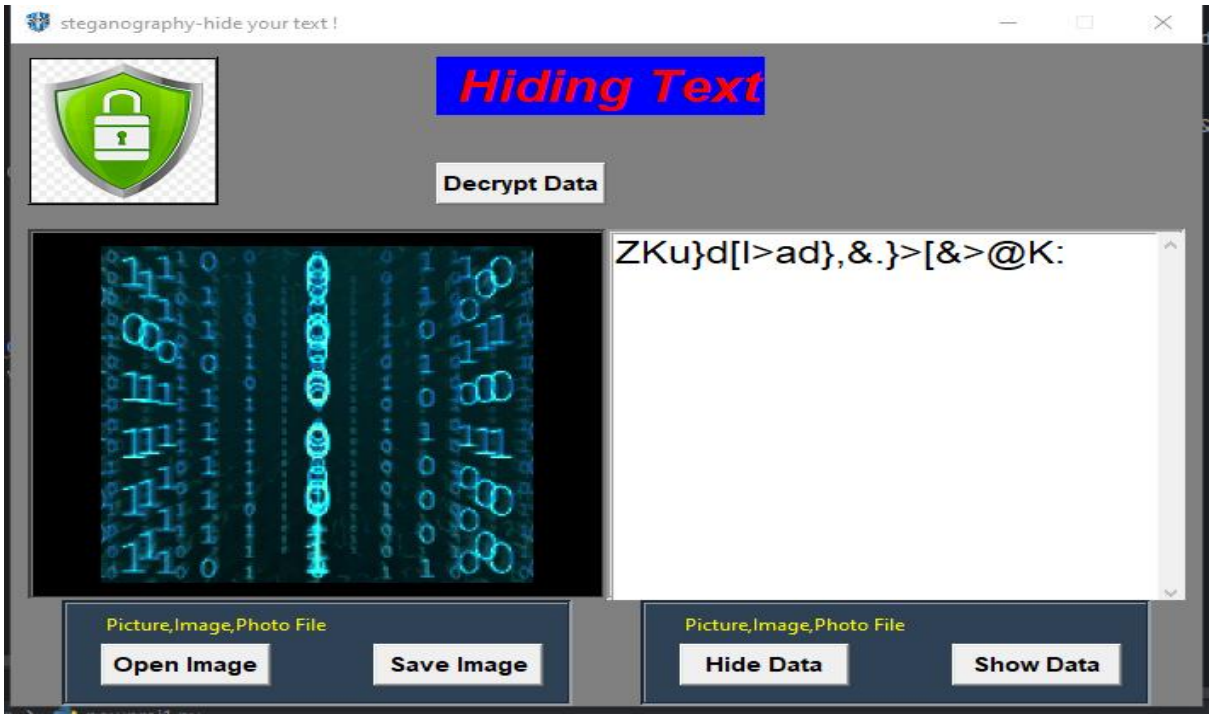


Figure:11

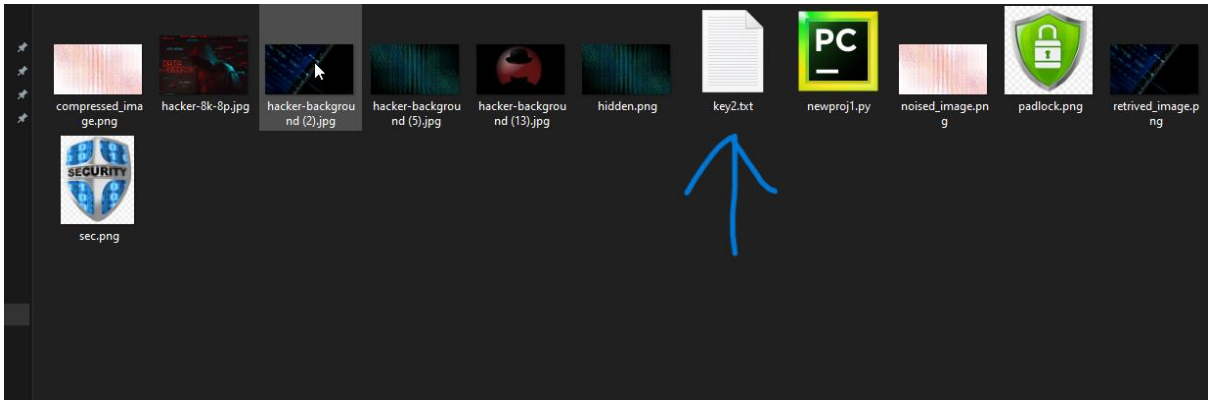


Figure:12

Figure:13

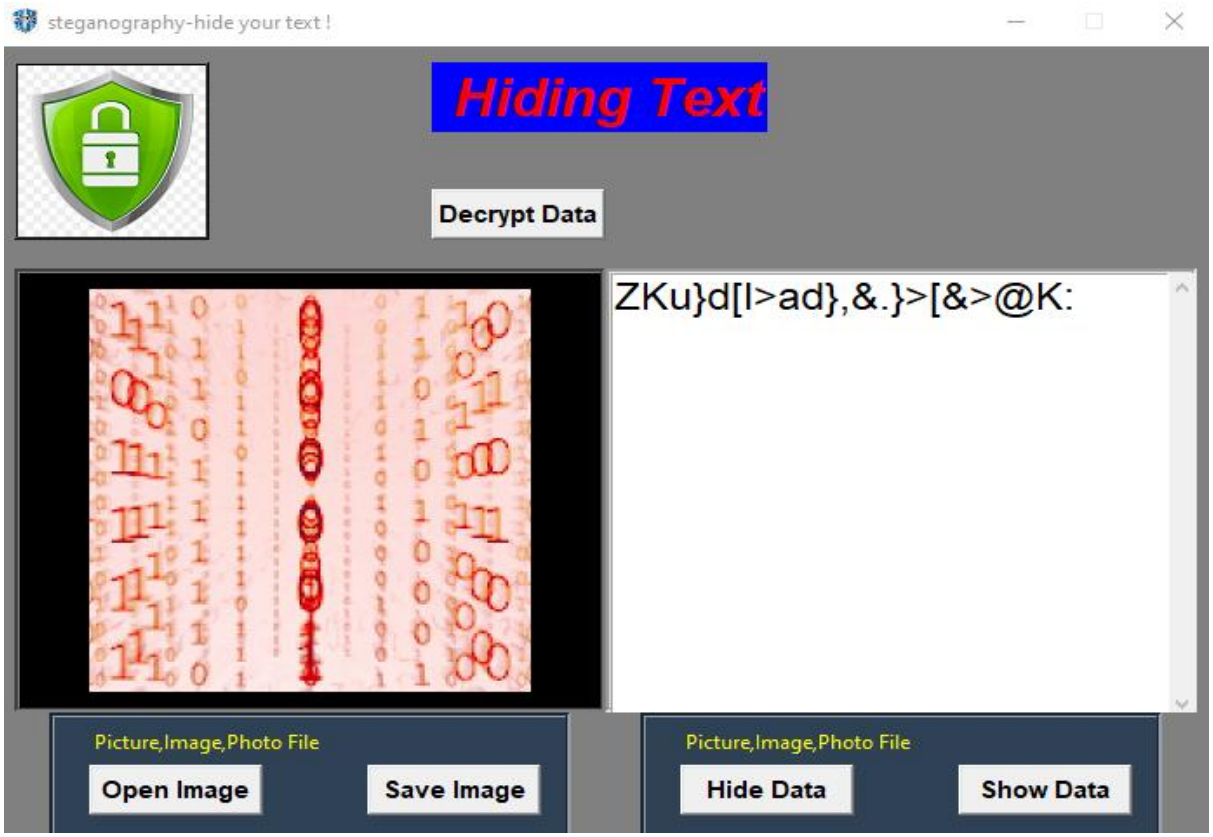


Figure:14

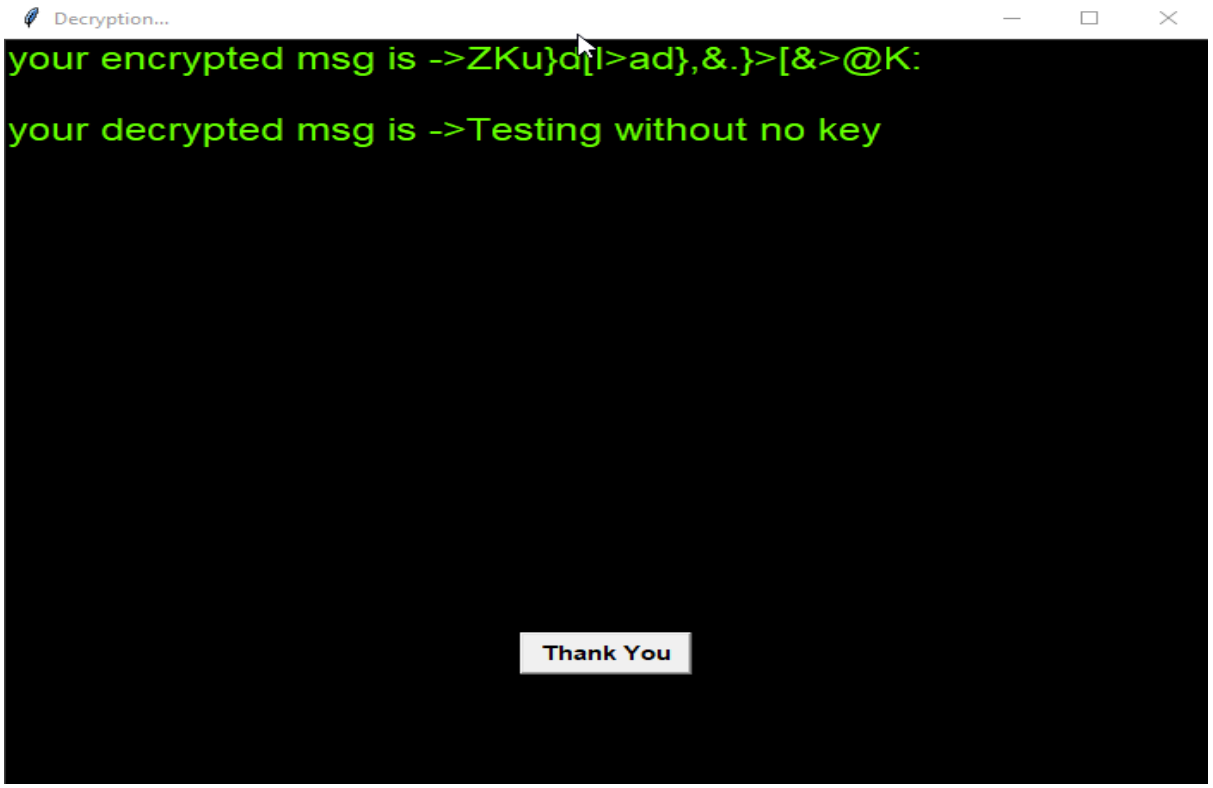


Figure:15

Chapter -7: Future Scope

The future of digital photography is promising and potential, thanks to the continuous advancement of technology and the changing environment of digital communication. In the future, many important areas will emerge that will present exciting opportunities for research, development, and application in the field of diagnostic imaging.

One of the best ways to explore the future is to incorporate deep learning techniques. The use of convolutional neural networks (CNNs) and generative adversarial networks (GANs) to analyze images is revolutionizing the field. These advanced neural network architectures have shown remarkable capabilities in pattern recognition and clustering, and their integration can improve the robustness and security of steganography techniques. Researchers can increase the level of sophistication of secret information injection by exploring neural networks to recognize and generate highly complex and secure steganographic patterns.

Enemy attacks and defenses are another important part of exploration. As steganography technology advances, so do the techniques hackers use to find and compromise hidden information. To ensure the effectiveness and security of these technologies, it is important to investigate enemy attacks on steganography systems and develop corresponding defenses. Developing robust countermeasures to evolving detection methods is essential to the longevity of steganography in secure communications.

The future of imaging involves exploring dynamic and adaptive technologies. Developing a steganography technique that can adapt to different image conditions and avoid detection is a major challenge. The adaptability of steganographic algorithms to different situations and environments is important to maintain efficiency in a wide range of applications.

A multi-domain approach to steganography can open new avenues for innovation. Researchers can explore hybrid methods that combine spatial and frequency methods, taking advantage of the unique strengths of each region. This approach can lead to steganography algorithms that are resistant to various detection methods.

Chapter -8: Conclusion & Reference

8.1 Conclusion:-

In conclusion, imaging is a dynamic and growing field with important implications for communications and information protection. Technological advances and new challenges in data security and privacy offer great opportunities for further research and development in this area.

A historical overview of image steganography shows the evolution from classical to modern methods. Early approaches such as LSB substitution paved the way for more complex algorithms, including spatial and frequency domain techniques. Current trends show the increasing integration of deep learning and the increased security and complexity of confidential information.

Security considerations are important, and tracking enemy attacks and how to protect against them shows the need for threat resilience. As steganography techniques become more sophisticated, it is important to develop robust countermeasures to avoid detection and compromise information.

The scope of image steganography includes a variety of interesting approaches. The introduction of deep learning technologies such as CNN and GAN provides an opportunity to revolutionize the field by improving the complexity and security of steganography techniques. Integration with real-time applications, quantum steganography and blockchain technology means that steganography is adapting to new technologies and paradigms. The adoption of dynamic and adaptive methods reflects the understanding that steganography techniques must be flexible and context-aware to handle a variety of situations. A multi-domain approach combining spatial and frequency domains offers the potential for more robust encryption techniques.

Biometric data hiding and privacy protection applications in distributed networks show the social relevance of image recognition. As data security and privacy concerns increase, steganography has become a valuable tool for protecting sensitive information while enabling communication. To obtain a systematic and comprehensive understanding of steganography technology, it is important to establish standard measures and evaluation. Standard frameworks contribute to the maturation of the field by establishing standardization and adoption of best practices. The future will create an environment where steganography will not only meet data security requirements, but will play an important role in creating a way to hide, communicate and protect sensitive information in the connected digital world.

8.2 Reference:

- https://en.wikipedia.org/wiki/Steganography
- https://www.kaspersky.com/resource-center/definitions/what-is-steganography
- https://www.researchgate.net/publication/348769709_Image_Steganography_A_Review_of_the_Recent_Advances
- https://www.geeksforgeeks.org/image-based-steganography-using-python/
- Pankaj jalote an overview of Project management "Software Project Management inPractice**"** Addison-Wesley,2002
- K.K. Aggarwal an overview of Project management "Software Engineering" New AgeInternational, 2005.

Annexure

Coding:

```
from tkinter import *
import cv2
from numpy import *
from tkinter import filedialog
import tkinter as tk
from PIL import Image, ImageTk
import os
from stegano import lsb

dict="aBb!@R#.CQcZdVeOfYgP)&Sh,]i-
Aj[k+T=D_}lm%U^'nEo{p:qF\\*(;r|WGs<KH>tuXvLJw?~MxyNz$"+'"'+" "

print(dict)

def encrypt(msg, key):
    try:
        encmsg = ""
        for i in msg:
            loc = key + dict.index(i)
            loc %= 82
            encmsg += dict[loc]
        print(encmsg)
        return encmsg
    except Exception as e:
        print(f"Error: {e}")



def decrypt(c, m):
    try:
        decmsg = ""
        for i in c:
            if i in dict:
                loc = dict.index(i) - m
                loc = (loc + len(dict)) % len(dict)
                decmsg += dict[loc]
            else:
                # Append characters not found in dict unchanged
                decmsg += i

        # print("here's your msg->", decmsg)
```

```
            return decmsg
        except Exception as e:
            print(f"Error: {e}")

    def Rndmkeygnrtr():
        key= list(dict)
        random.shuffle(key)
        return (''.join(key))

    def encrypt2(x,ky):
        try:
            encmsg=""
            charsA=dict
            charsB=ky
            for i in x:
                index=charsA.find(i)
                encmsg+=charsB[index]
            print('\nencryption is complete')
            print('encrypted text is->',encmsg)
            return encmsg
        except Exception as e:
            print(f"Error: {e}")

    def decrypt2(x,h) :
        try:
            dcmsg=""
            charsA = dict
            charsB = h
            charsA,charsB=charsB,charsA
            for i in x:
                index = charsA.find(i)
                dcmsg += charsB[index]
            print('\ndecryption is complete')
            print("decrypted text is->",dcmsg)
            return dcmsg
        except Exception as e:
            print("error is:",e)
    def getval():
        try:
            print(f"msg is {msg.get()}")
            print(f"key is {key.get()}")
            x = msg.get()
            y = key.get()
            if y!=0:
               z= encrypt(x,y)
               return z
            if y==0:
                fout = open("key2.txt", 'w')
```

```
            key2 = Rndmkeygnrtr()
            fout.write(key2)
            # ky=fout.read()
            print("ky is",key2)
            z=encrypt2(x,key2)
            return z
    except Exception as e:
        print(f"Error: {e}")


def add_one_to_bits(image):
    try:
        # Add 1 to each binary bit of each pixel
        noisy_image = cv2.bitwise_not(image)
        return noisy_image
    except Exception as e:
        print(f"Error: {e}")



def retrieve_original(image):
    try:
        # Retrieve the original image by applying the same operation
        original_image = cv2.bitwise_not(image)
        return original_image
    except Exception as e:
        print(f"Error: {e}")


def noise():
    try:
        # Load an image from file
        image_path = 'hidden.png'
        original_image = cv2.imread(image_path)
        # Add 1 to each binary bit of each pixel in the image
        noisy_image = add_one_to_bits(original_image)
        # Display the noisy images
        cv2.imwrite("noised_image.png", noisy_image)
        input_image_path = "noised_image.png"
        compressed_image_path = "compressed_image.png"
        compress_image(input_image_path, compressed_image_path)
    except Exception as e:
        print(f"Error: {e}")

def denoise():
    try:
        image_path = 'decompressed_image.png'
        noisy_image = cv2.imread(image_path)
        # Retrieve the original image from the noisy image
```

```
            retrieved_image = retrieve_original(noisy_image)

            # Display the retrieved image
            cv2.imwrite("retrived_image.png", retrieved_image)
            Show()
            cv2.destroyAllWindows()
        except Exception as e:
            print(f"Error: {e}")


    def rootclose():
        try:
            root.quit()
            fin.quit()
        except Exception as e:
            print(f"Error: {e}")

    def compress_image(input_path, output_path, quality=30):
        try:
            # Open the image file
            img = Image.open(input_path)

            # Save the compressed image
            img.save(output_path, quality=quality)

            print(f"Image compressed successfully and saved to {output_path}")
        except Exception as e:
            print(f"Compression Error: {e}")

    def decompress_image():
        try:
            input_path = "compressed_image.png"

            # Open the compressed image file
            img = Image.open(input_path)
            output_path = "decompressed_image.png"
            # Save the decompressed image
            img.save(output_path)

            print(f"Image decompressed successfully and saved to {output_path}")
            denoise()
        except Exception as e:
            print(f"Decompression Error: {e}")


    win = Tk()

    win.geometry("800x300")
    usr = Label(win, text='enter your msg')
```

```
psd = Label(win, text='key')
usr.grid(row=0, column=1)
psd.grid(row=1, column=1)

msg = StringVar()
key = IntVar()

msgentry = Entry(win, textvariable=msg)
keyentry = Entry(win, textvariable=key)

msgentry.grid(row=0, column=10)
keyentry.grid(row=1, column=10)
b = Button(win, text='submit')
b.grid(row=2, column=1)

b = Button(win, text='close', command=win.destroy)
b.grid(row=2, column=2)
win.mainloop()


# functions
def showimage():
    try:
        global filename
        filename=filedialog.askopenfilename(initialdir=os.getcwd(),title="sele
ct image file",
                                            filetype=(("JPG
file","*.jpg"),("PNG file","*.png"),("all file","*.*")))
        img=Image.open(filename)
        img=ImageTk.PhotoImage(img)
        lbl.configure(image=img,width=250,height=250)
        lbl.image=img
    except Exception as e:
        print(f"Error: {e}")


def Hide():
    try:
        global secret
        message = text1.get(1.0, END)
        secret = lsb.hide(str(filename), message)
    except Exception as e:
        print(f"Error: {e}")


def Show():
    try:

        clear_message = lsb.reveal("retrived_image.png")
```

```
            # Get the key from the key entry
            m = key.get()
            # Update the text area with the revealed and decrypted message
            text1.delete(1.0, END)
            text1.insert(1.0, clear_message)

            return clear_message
        except Exception as e:
            print(f"Error: {e}")

    def save():
        try:
            s= secret.save("hidden.png")
            noise()

        except Exception as e:
            print(f"Error: {e}")

    # 2nd interface

    root = Tk()
    root.title("steganography-hide your text !")
    root.geometry("700x500+150+170")
    root.resizable(False, False)
    root.configure(bg="grey")
    # icon
    imageicon = PhotoImage(file="sec.png")
    root.iconphoto(False, imageicon)

    # logo
    logo = PhotoImage(file="padlock.png")
    Label(root, image=logo, bg="black", relief=RAISED).place(x=10, y=10)
    Label(root, text=" Hiding Text", bg="blue", fg="red", font="arial 25 italic
    bold").place(x=250, y=10)

    # 1st frame
    f = Frame(root, bd=3, bg='black', width=340, height=280, relief=GROOVE)
    f.place(x=10, y=140)
    lbl = Label(f, bg="black")
    lbl.place(x=40, y=10)

    # 2nd frame
    frame2 = Frame(root, bd=3, width=340, height=280, bg="white", relief=GROOVE)
    frame2.place(x=350, y=140)
    text1 = Text(frame2, font="arial 18", fg="black", relief=GROOVE)
    text1.place(x=0, y=0)
    if len(msg.get()) != 0 :
```

```
    x = str(getval())
    text1.insert(1.0, x)
# text1.config(state= DISABLED)
scrollbar1 = Scrollbar(frame2)
scrollbar1.place(x=320, y=0, height=280)
scrollbar1.configure(command=text1.yview)
text1.configure(yscrollcommand=scrollbar1.set)

# third frame
frame3 = Frame(root, bd=3, bg="#2f4155", width=300, height=80, relief=GROOVE)
frame3.place(x=30, y=420)
Button(frame3, text="Open Image", width=10, height=1, font="arial 11 bold",
command=showimage).place(x=20, y=30)
Button(frame3, text="Save Image", width=10, height=1, font="arial 11 bold",
command=save).place(x=180, y=30)
Label(frame3, text="Picture,Image,Photo File", bg="#2f4155",
fg="yellow").place(x=20, y=5)

# forth frame
frame4 = Frame(root, bd=3, bg="#2f4155", width=300, height=80, relief=GROOVE)
frame4.place(x=370, y=420)
Button(frame4, text="Hide Data", width=10, height=1, font="arial 11 bold",
command=Hide).place(x=20, y=30)
Button(frame4, text="Show Data", width=10, height=1, font="arial 11 bold",
command=decompress_image).place(x=180, y=30)

Label(frame4, text="Picture,Image,Photo File", bg="#2f4155",
fg="yellow").place(x=20, y=5)

Button(root, text="Decrypt Data", width=10, height=1, font="arial 11 bold",
command=rootclose).place(x=250, y=90)

root.mainloop()



#third frame
fin=Tk()
fin.geometry("700x600")
fin.title("Decryption...")
text4 = Text(fin, font="arial 18", fg="#66FF00",bg="black" ,relief=GROOVE)
text4.place(x=0, y=0)
encrypted_message = Show()
m = key.get()

if m!=0:
    dec = decrypt(encrypted_message, m)
if m==0:
    try:
```

```
            k = open("key2.txt", "r")
            h = k.read()
            dec = decrypt2(encrypted_message,h)
        except Exception as e:
            print("error is:", e)


text4.insert(1.5,f'your encrypted msg is ->{encrypted_message}')
text4.insert(2.5,f'\nyour decrypted msg is ->{dec}')
Button(fin, text="Thank You", width=10, height=1, font="arial 11 bold",
command=rootclose).place(x=300, y=450)
fin.mainloop()
```