A Minor Project

on

HIDE SECRET TEXT MESSAGE INSIDE IMAGE USING PYTHON | STEGANOGRAPHY |

Submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD

In Partial fulfilment of the requirement for the award of degree of

BACHELOR OF TECHNOLOGY IN

COMPUTER SCIENCE AND ENGINEERING

By

R. KISHORE	**- 21RA1A0549**
K. DINESHKUMAR	**- 21RA1A0528**
V.LAXMANREDDY	**- 21RA1A0550**

Under the guidance of

Mrs. E. Sravanthi

Professor, CSE Dept.

Department of Computer Science and Engineering

- DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
- KOMMURI PRATAP REDDY INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUH, Ghanpur(V), Ghatkesar(M), Medchal(D)-500088)

2021 -2025

KOMMURI PRATAP REDDY INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUH, Ghanpur(V), Ghatkesar(M), Medchal(D)-500088)

CERTIFICATE

This is to certify that the project work entitled **“Hide Secret Text Message Inside Image Using Python | Steganography |”** is submitted by Mr.R. KISHORE , Mr.K. Dinesh kumar, Mr.v. Laxman reddy Bonafide students of **Kommuri Pratap Reddy Institute of Technology** in partial fulfilment ofthe requirement for the award of the degree of Bachelor of Technology in **Computer Science and Engineering** of the **Jawaharlal Nehru Technological University Hyderabad,** during the year 2024-25.

Internal Guide
Mrs. E Sravanthi

HOD
Dr. S. Kavitha

Project Coordinator
Mr. P. vijay

External Examiner

DECLARATION

We hereby declare that this project work entitled **“Hide Secret Text Message Inside Image Using Python | Steganography |”** in partial fulfilment of requirements for the award of degree of Computer Science and Engineering is a Bonafide work carried out by us during the academic year 2024- 2025

We further declare that this project is a result of our effort and has not been submitted for the award ofany degree by us to any institute.

By

R. KISHORE	**(21RA1A0549)**
K. DINESH KUMAR	**(21RA1A0528)**
V.LAXMAN REDDY	**(21RA1A0550)**

ACKNOWLEDGEMENT

It gives us immense pleasure to acknowledge with gratitude, the help and support extended throughout the project report from the following:

We will be very much grateful to almighty our **Parents** who have made us capable of carrying out our job.

We express our profound gratitude to Dr.RAVINDRA EKLARKER, Principal of Kommuri Pratap Reddy Institute of Technology, who has provided necessary infrastructure and resources in completing our project report successfully.

We are grateful to Dr. S. Kavitha who is our Head of the Department, CSE for her amiable ingenious and adept suggestions and pioneering guidance during the project report.

We express our gratitude and thanks to the coordinator MR.P. VIJAY of our department for her contribution for making it success within the given time duration.

We express our deep sense of gratitude and thanks to Internal Guide, MRS.E SRAVANTHI Professor for his guidance during the project.

We are also very thankful to our **Management, Staff Members** and all **Our Friends** for their valuable suggestions and timely guidance without which we would not have been completed it.

By

R. KISHORE	**(21RA1A0549)**
K. DINESH KUMAR	**(21RA1A0528)**
V. LAXMAN REDDY	**(21RA1A0550)**

Vision of the Institute

To emerge as a premier institute for high quality professional graduates who can contribute to economic and social developments of the Nation.

Mission of the Institute

Mission	Statement
IM1	To have holistic approach in curriculum and pedagogy through industry interface to meet the needs of Global Competency.
IM2	To develop students with knowledge, attitude, employability skills, entrepreneurship, research potential and professionally Ethical citizens.
IM3	To contribute to advancement of Engineering & Technology that would help to satisfy the societal needs.
IM4	To preserve, promote cultural heritage, humanistic values and Spiritual values thus helping in peace and harmony in the society.

Vision of the Department

To Provide Quality Education in Computer Science for the innovative professionals to work for the development of the nation.

Mission of the Department

Mission	Statement
DM1	Laying the path for rich skills in Computer Science through the basic knowledge of mathematics and fundamentals of engineering
DM2	Provide latest tools and technology to the students as a part of learning infrastructure
DM3	Training the students towards employability and entrepreneurship to meet the societal needs.
DM4	Grooming the students with professional and social ethics.

Program Educational Objectives (PEOs)

PEO's	Statement
PEO1	The graduates of Computer Science and Engineering will have successful career in technology.
PEO2	The graduates of the program will have solid technical and professional foundation to continue higher studies.
PEO3	The graduate of the program will have skills to develop products, offer services and innovation.

PEO4	The graduates of the program will have fundamental awareness of industry process, tools and technologies.

Program Outcomes

PO1	**Engineering Knowledge:** Apply the knowledge of mathematics, science, Engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO2	**Problem Analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	**Design/development of Solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	**Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	**Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO6	**The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO7	**Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental context, and demonstrate the knowledge of, and need for sustainable development.
PO8	**Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9	Individual and team network: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, being able to comprehend and write effective reports and design documentation, make Effective presentations, and give and receive clear instructions.
PO11	Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work,as a member and leader in a team, to manage projects and in multidisciplinary environment.
PO12	Life-Long learning: Recognize the need for, and have the preparation and able to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES

PSO1	Foundation of mathematical concepts: To use mathematical methodologies to crack problem using suitable mathematical analysis, data structure and suitable algorithm.
PSO2	Foundation of Computer Science: The ability to interpret the fundamental concepts and methodology of computer systems. Students can understand the functionality of hardware and software aspects of computer systems.
PSO3	Foundation of Software development: The ability to grasp the software development lifecycle and methodologies of software systems. Possess competent skills and knowledge of software design process.

TABLE OF CONTENTS

LIST OF FIGURES

TABLE OF SCREENSHOTS

ABSTRACT

In an age of increasing digital communication and data transfer, ensuring the security and privacy of sensitive information is paramount. Steganography, the art of hiding information within other data, has been used for centuries. In the digital realm, it plays a critical role in secure communication and information concealment. Traditional steganography methods often involve embedding information within a single image. While effective, this approach may be susceptible to detection, as single-image steganography can leave detectable traces, especially under sophisticated analysis. The primary challenge is to develop a robust system for multiple image steganography that can securely hide sensitive files within a set of images. This involves designing algorithms that distribute the information effectively across the images while maintaining imperceptibility and ensuring reliable extraction. Therefore, the rise of cyber threats and privacy concerns, there's a growing need for advanced techniques to protect sensitive files from unauthorized access or interception. Multiple image steganography, an emerging field, offers the potential for heightened security by spreading information across multiple images, making it even more challenging for potential adversaries to detect or extract. The project seeks to enhance file security by leveraging advanced techniques in multiple image steganography. By distributing the information across a set of images, this research endeavors to develop a system capable of securely concealing sensitive files. The algorithms utilized in this approach are designed to ensure imperceptibility and robustness against detection efforts. This advancement holds great promise for significantly improving the security of file transmission and storage, safeguarding critical information from unauthorized access or interception.

CHAPTER 1

INTRODUCTION

1.1 History

In today's digital era, safeguarding sensitive information during communication and data transfer is of utmost importance. Throughout history, the practice of steganography, which involves concealing information within other data, has been utilized to achieve this goal. As we transition to digital mediums, steganography plays a crucial role in ensuring secure communication and information concealment.

Traditionally, steganography focused on embedding information within a single image. While effective, this approach has its drawbacks, as it can be susceptible to detection, especially under sophisticated analysis. The challenge at hand is to develop a robust system for multiple image steganography that can securely hide sensitive files within a collection of images. This entails designing algorithms that efficiently distribute the information across the images while maintaining imperceptibility and ensuring reliable extraction.

The increasing prevalence of cyber threats and privacy concerns underscores the need for advanced techniques to protect sensitive files from unauthorized access or interception. Multiple image steganography, an emerging field, offers the potential for heightened security by dispersing information across multiple images. This makes it even more challenging for potential adversaries to detect or extract the concealed data.

The project, titled "Elevating file security through advances in multiple image steganography," aims to enhance file security by leveraging advanced techniques in multiple image steganography. The goal is to develop a system capable of securely concealing sensitive files by distributing information across a set of images. The algorithms employed in this approach are specifically designed to ensure imperceptibility and robustness against detection efforts. This advancement holds great promise for significantly improving the security of file transmission and storage, safeguarding critical information from unauthorized access or interception.

1.2 Problem Statement

In the current era of digital communication and data transfer, the protection of sensitive information is a critical concern. Steganography, a practice with historical roots, involves concealing data within other information and has become increasingly relevant in the digital landscape for ensuring secure communication and information protection.

Traditional steganography methods have typically focused on embedding information within a single image. While effective, this approach is not foolproof and can be vulnerable to detection, particularly under sophisticated analysis. The primary challenge lies in developing a robust system for multiple image steganography, where sensitive files can be securely hidden within a group of images. This task requires the creation of algorithms that can distribute information effectively across these images while maintaining imperceptibility and enabling reliable extraction.

The growing prevalence of cyber threats and concerns about privacy highlight the necessity for advanced techniques to safeguard sensitive files from unauthorized access or interception. Multiple image steganography, an emerging field, offers the potential for enhanced security by dispersing information across several images. This complexity makes it considerably more challenging for potential adversaries to detect or extract concealed data.

The objective of the project, titled "Elevating file security through advances in multiple image steganography," is to improve file security by harnessing advanced techniques in multiple image steganography. The aim is to create a system capable of securely concealing sensitive files by distributing information across a set of images. The algorithms employed in this approach are meticulously designed to ensure imperceptibility and robustness against detection efforts. This advancement holds significant promise for enhancing the security of file transmission and storage, protecting critical information from unauthorized access or interception.

1.3 Research Motivation

In the contemporary landscape of digital communication and information exchange, the safeguarding of sensitive data is a paramount concern. As we navigate an increasingly interconnected world, the need for robust security measures to protect confidential information becomes more critical than ever. The motivation for this research stems from the realization that traditional steganography methods, particularly those centered around embedding information within a single image, may fall short in providing foolproof security.

Recognizing the potential vulnerabilities associated with single-image steganography, the research seeks to address a pressing challenge: the development of an advanced system for multiple image steganography. The primary motivation lies in the aspiration to create a more secure environment for concealing sensitive files within a collection of images. Current methodologies, while effective to some extent, may leave detectable traces that could be exploited by sophisticated analysis.

The escalating threat landscape of cyber-attacks and the heightened concerns surrounding privacy underscore the urgency of advancing our security techniques. In this context, multiple image steganography emerges as a promising avenue. By distributing information across a series of images, the research aims to significantly elevate the complexity of concealing sensitive data, making it notably more challenging for potential adversaries to detect or extract such information.

The overarching motivation is to contribute to the enhancement of file security by leveraging cutting-edge techniques in multiple image steganography. The ultimate goal is to develop a system that not only ensures the imperceptibility of concealed information but also exhibits robustness against detection efforts. This research is driven by the imperative to fortify the security of file transmission and storage, thereby safeguarding critical information from unauthorized access or interception in an ever-evolving digital landscape.

1.4 Applications

- **Secure Communication in Sensitive Environments:**

  In government and military communications, where the utmost secrecy is crucial, the proposed multiple image steganography techniques can be employed to transmit classified information securely. By dispersing the data across a set of images, the communication becomes more resistant to unauthorized interception or detection.

- **Confidential Data Transfer in Healthcare:**
  In the healthcare sector, where the exchange of sensitive patient data is routine, the enhanced security provided by multiple image steganography can be invaluable. Medical records, research findings, and other confidential information can be securely transmitted, reducing the risk of data breaches.

- **Financial Data Protection:**

  Financial institutions handling sensitive financial transactions and client information can benefit from the heightened security offered by this research. Multiple image steganography can be integrated into financial systems to safeguard sensitive data, such as transaction details, account information, and proprietary algorithms.

- **Legal and Corporate Document Protection**:

  Law firms and corporate entities dealing with confidential legal documents, contracts, and proprietary information can use these techniques to bolster document security. Multiple image steganography ensures that critical information remains discreet and resistant to unauthorized access or intellectual property theft.

- **Journalism and Whistleblower Protection:**

  Journalists and whistleblowers seeking to protect their sources and sensitive information can employ multiple image steganography for secure communication. This application is particularly relevant in regions or situations where freedom of the press is under threat, ensuring the safety of those involved.

- **Digital Rights Management (DRM):**

  In the digital entertainment industry, protecting intellectual property, such as copyrighted videos or images, is paramount. Multiple image steganography can contribute to robust digital rights management systems, preventing unauthorized copying or distribution of multimedia content.

- **Research and Development Security:**

  Companies involved in research and development can use these techniques to safeguard proprietary technologies, product designs, and research findings. Multiple image steganography adds an additional layer of security, making it more challenging for competitors or malicious entities to gain access to critical information.

CHAPTER 2

LITERATURE SURVEY

B. Sultan,et.al[1] In this project The success of deep learning based steganography has shifted focus of researchers from traditional steganography approaches to deep learning based steganography. Various deep steganographic models have been developed for improved security, capacity and invisibility. In this work a multi-data deep learning steganography model has been developed using a well known deep learning model called Generative Adversarial Networks (GAN) more specifically using deep convolutional Generative Adversarial Networks (DCGAN). The model is capable of hiding two different messages, meant for two different receivers, inside a single cover image. The proposed model consists of four networks namely Generator, Steganalyzer Extractor1 and Extractor2 network. The Generator hides two secret messages inside one cover image which are extracted using two different extractors. The Steganalyzer network differentiates between the cover and stego images generated by the generator network. The experiment has been carried out on CelebA dataset. Two commonly used distortion metrics Peak signal-to-Noise ratio (PSNR) and Structural Similarity Index Metric (SSIM) are used for measuring the distortion in the stego image The results of experimentation show that the stego images generated have good imperceptibility and high extraction rates.

X. Liao,et.al[2] In this project With the coming era of cloud technology, cloud storage is an emerging technology to store massive digital images, which provides steganography a new fashion to embed secret information into massive images. Specifically, a resourceful steganographer could embed a set of secret information into multiple images adaptively, and share these images in cloud storage with the receiver, instead of traditional single image steganography. Nevertheless, it is still an open issue how to allocate embedding payload among a sequence of images for security performance enhancement. This article formulates adaptive payload distribution in multiple images steganography based on image texture features and provides the theoretical security analysis from the steganalyst's point of view. Two payload distribution strategies based on image texture complexity and distortion distribution are designed and discussed, respectively. The proposed strategies can be employed together with these state-of-the-art single image steganographic algorithms. The comparisons of the security performance against the modern universal pooled steganalysis are given. Furthermore, this article compares the per image detectability of these multiple images steganographic schemes against the modern single image steganalyzer. Extensive experimental results show that the proposed payload distribution strategies could obtain better security performance.

M. Srivastava,et.al[3] In this project Image Steganography is the artwork of concealing mystery data within the image such that the hacker will now no longer be capable of discover the records within inside the stego images. This is a useful approach to secure our sensitive information. Security has continually been a main difficulty from last many years to existing days. The topic of interest to researchers has long been the development of secure technologies for sending data to anyone other than the recipient without revealing it. Therefore, from nowadays, researchers have evolved many strategies to meet the steady transfer of information and steganography is one in all them. In this paper, we work on two techniques for hiding information in the image. First, we do analysis on LSB for storing information bit. As the technique is known to all, the attacker will be able to easily reveal the information, this makes image steganography unsecured. Secondly, R-Color Channel encoding with RSA set of rules for offering extra protection to information in addition to our information hiding approach. The proposed approach makes use of a red color channel for hiding information bits and the following bits for RGB pixel values of the original image. This paper present the performance analysis of two most popular algorithms, LSB and RSA along with image steganography.

G. Benedict,et.al[4] In this project Steganography is the process of hiding a secret message within an ordinary message & extracting it at its destination. Image steganography is one of the most common and secure forms of steganography available today. Traditional steganography techniques use a single cover image to embed the secret data which has few security shortcomings. Therefore, batch steganography has been adopted which stores data on multiple images. In this paper, a novel approach is proposed for slicing the secret data and storing it on multiple cover images. In addition, retrieval of this secret data from the cover images on the destination side has also been discussed. The data slicing ensures secure transmission of the vital data making it merely impossible for the intruder to decrypt the data without the encrypting details.

S. Mukhopadhyay,et.al[5] In this The paper proposes a scheme for achieving steganography with multiple encrypted monochromatic images with keys obtained from a synchronized system of semiconductor lasers. The key selection scheme for steganography determines the robustness of the application. It is in this area that steganography may benefit from the properties of chaos synchronization. The encryption principle of the new algorithm is analyzed quantitatively by various statistical tests. The cover image used in the technique is also obtained from the visual representation of the chaotic sequences. This new scheme enjoys the benefit of added security, high key space, high embedding capacity, imperceptibility and robustness of the hidden information in conjunction with Least Significant Bit (LSB) based substitution. The result is important from the perspective of introducing a mechanism to multiplex and simultaneously transmit multiple images.

A. S. Ansari,et.al[6] This paper presents an image Steganography algorithm that can work for cover images of multiple formats. Having a single algorithm for multiple image types provides several advantages. For example, we can apply uniform security policies across all image formats, we can adaptively select the most suitable cover image based on data length, network bandwidth and allowable distortions, etc. We present our algorithm based on the abstract concept of image components that can be adapted for JPEG, Bitmap, TIFF and PNG cover images. To the best of our knowledge, the proposed algorithm is the first Steganography algorithm that can work for multiple cover image formats. In addition, we have utilized concepts like capacity pre-estimation, adaptive partition schemes and data spreading to embed secret data with enhanced security. The proposed method is tested for robustness against Steganalysis with favorable results. Moreover, comparative results for the proposed algorithm are very promising for three different cover image formats.

P. Grandhe,et.al[7] In this project Communicating online without fearing third-party interventions is becoming a challenge in the modern world. Especially the sectors like the military, and government organizations or private companies sharing sensitive information. They invest a lot of effort and cost into obtaining the advancement of safe communication techniques. Image processing encryption techniques using various algorithms promote security over communication channels and using different analysis methods make the tool stand out in providing security to the information. In today's world, there are various steganographic mechanisms that convert the secret message into stego medium and send it across various communication channels. Using algorithms like Blind Hide promotes the security of the message along with using multiple analysis methods that will further improve the tool in giving out information of encoded accuracy, size of stego of the secret message. The aim is to generate a tool that will give out a benchmark value of how precisely the message is stored in the cover file. Using Stego and bulk analysis the information about the presence of the stego medium in the message can be known to the user. All these analysis methods make the tool more enhanced and secure.

R. Joshi,et.al[8] in this There are advances in data stealth and forgery with the rise of technology and advances in data transmission. This arises the need for better and developed methods for data transmission. Data Transmission is an essential task in the current era, and equally important is the secure and safe information of that data. In the paper, batch steganography is used to secure data transmission from one end to the other. Often a password can be used for encoding the payload into the cover image. Here the data is encrypted using hashing and encryption techniques, SHA-256 and AES. The passwords used for encryption have been used after the logical operation XOR. Thus, the information has been encrypted twice using the XORed password for first and second input password for the next time. It increases the security of the data and makes the decryption almost impossible without knowing both the passwords and the encryption method. The encoded data is then embedded within the pixels of the original image using the LSB method. This prevents data theft and any possibilities of Man-in-the-Middle attacks since the time required for decrypting the data is drastically high without the knowledge of the inputs and the techniques used.

Z. Wang,et.al[9] In this paper, a more accurate image steganography method is proposed, where a multi-level feature fusion procedure based on GAN is designed. Firstly, convolution and pooling operations are added to the network for feature extraction. Then, short links are used to fuse multi-level feature information. Finally, the stego image is generated by confrontation learning between discriminator and generator. Experimental results show that the proposed method has higher steganalysis security under the detection of high-dimensional feature steganalysis and neural network steganalysis. Comprehensive experiments show that the performance of the proposed method is better than ASDL-GAN and UT-GAN.

X. Zhao,et.al[10]In this paper, a multi dilated generation countermeasure network (Multi Dilated GAN) model is proposed to improve the information steganography quality of images. Based on the discriminator of steganogan model, multiple convolution and expansion convolution are adopted. By selecting different expansion rates in the expansion convolution and matching with different receptive fields, different feature layers of the network can be effectively extracted; Multiple convolution processing is carried out on different feature layers to connect the features extracted from different directions, and distinguish the steganographic image from the carrier image, so that the discriminator can more accurately distinguish the difference between the steganographic image and the carrier image, so as to improve the steganographic ability of the encoder of the model.

Experiments show that this model can effectively improve the steganogan model's steganographic accuracy and steganographic quality while maintaining a high steganographic capacity of 4.4bit/pixel.

M. Liu,et.al[12] In this paper, They propose a new adversarial embedding scheme for image steganography. Unlike those related works, we first combine multiple gradients of cover and generated stegos to determine the directions of cost modifications. Next, instead of adjusting all or a random part of embedding costs in existing works, we carefully select the candidate costs according to the amplitudes of cover gradients and their costs. Extensive experimental results demonstrate that by adjusting a tiny part of embedding costs (less than 5% in most cases), the proposed method can significantly improve the security of five modern steganographic methods evaluated on both re-trained CNN-based and traditional steganalysis, and achieve much better security performances compared with related methods. In addition, the security performances evaluated on different image database show that the generalization of the proposed method is good.

CHAPTER 3

EXISTING SYSTEM

3.1 Least Significant Bit (LSB) Substitution in Single Image Steganography

Steganography, the practice of concealing information within seemingly innocuous carriers, plays a crucial role in secure communication and data protection. Within the realm of steganographic techniques, Least Significant Bit (LSB) substitution stands out as a fundamental and widely employed method, particularly in the domain of single image steganography. Steganography aims to transmit hidden information in a manner that avoids detection by unintended recipients. Unlike cryptography, which focuses on rendering data unreadable, steganography is concerned with making the existence of the information inconspicuous. Images, being a ubiquitous form of digital content, serve as ideal carriers for covert communication through steganographic methods.At its core, LSB substitution leverages the binary representation of digital images. In an 8-bit grayscale image, each pixel is represented by 8 bits, ranging from 00000000 to 11111111. The least significant bit (the rightmost bit) of each pixel is the one that contributes the least to the overall pixel value. It is this bit that LSB substitution manipulates to embed hidden information. LSB substitution is revered for its simplicity and effectiveness. The idea is to replace the least significant bit of each pixel in a cover image with bits from the secret message. Since the changes occur in the least influential part of the pixel, they are generally imperceptible to the human eye. This simplicity makes LSB substitution a popular choice for scenarios where a balance between ease of implementation and modest data hiding capacity is sought.

3.2 Disadvantages of Least Significant Bit (LSB) Substitution

1. Vulnerability to Statistical Analysis:

One of the primary weaknesses of LSB substitution is its susceptibility to statistical analysis. Changes made to the least significant bits may introduce discernible patterns, especially in regions with uniform pixel values. Adversaries can exploit these patterns to detect the presence of hidden information, compromising the security of the steganographic process.

2. Limited Capacity for Data Hiding:

LSB substitution has a restricted capacity for hiding large volumes of data. Since only one bit per pixel is altered, the amount of information that can be embedded within an image is relatively modest. In scenarios requiring substantial data concealment, alternative steganographic methods with higher capacities may be more suitable.

3. Potential for Visual Artifacts:

Despite efforts to minimize the visual impact of LSB substitution, there remains a risk of introducing noticeable artifacts, especially in images with high contrast or in specific color channels of a color image. Changes to LSBs might result in subtle alterations that, while not easily perceptible, can potentially be detected through careful visual inspection.

4. Susceptibility to Compression:

LSB substitution may be affected by image compression algorithms, potentially leading to unintended alterations in the hidden data or a reduction in the overall quality of the stego image. Compression processes, such as JPEG compression, may discard or modify LSBs, impacting the reliability of the embedded information.

5. Lack of Robustness:

LSB substitution lacks robustness in the face of certain image processing operations. Operations like cropping, resizing, or filtering may affect the LSBs, potentially leading to the loss or corruption of hidden data. The fragility of LSB-based steganography under various transformations limits its reliability in real-world applications.

6. Detectability in High-Frequency Components:

In images with high-frequency components or detailed textures, modifications introduced by LSB substitution may become more noticeable. Adversaries exploiting frequency analysis techniques might focus on these regions to detect hidden information, reducing the overall security of the steganographic method.

7. Inability to Withstand Advanced Attacks:

LSB substitution may falter when exposed to advanced steganalysis techniques. As adversaries develop sophisticated algorithms and statistical models, the basic concealment provided by LSB substitution becomes less effective in thwarting detection, particularly in the face of determined attacks.

8. Randomization Challenges:

While randomized LSB substitution is a proposed enhancement to mitigate detectability, achieving true randomness in the selection of pixels for modification can be challenging. Pseudorandom sequences may still exhibit patterns, and ensuring a truly unpredictable distribution requires careful implementation.

9. Not Suitable for Continuous Tone Images:

LSB substitution is more suited for discrete-tone images where changes in pixel values are less likely to be visually apparent. In continuous-tone images, especially those with smooth gradients, alterations to LSBs may be more noticeable, compromising the overall concealment.

CHAPTER 4

PROPOSED SYSTEM

4.1 Overview

Step 1: Steganography Dataset

The initial step in our research involves acquiring a suitable dataset for steganography. This dataset consists of various images that will serve as carriers for the hidden information. The selection of these images is crucial, as their characteristics can influence the effectiveness and imperceptibility of the steganographic technique. Typically, a diverse set of images with varying resolutions, color distributions, and complexities are chosen to ensure the robustness and generalizability of the proposed method.

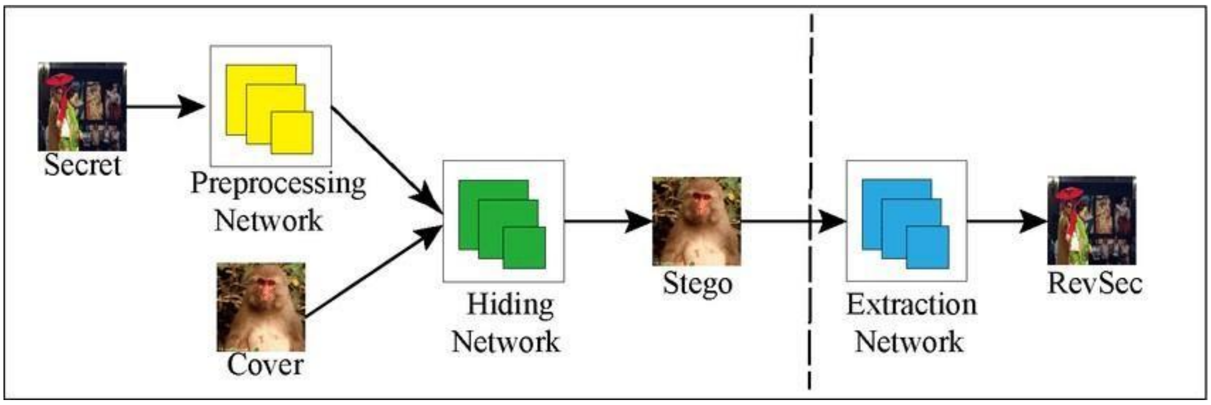


Fig. 4.1: Image steganography architecture.

Step 2: Dataset Preprocessing

Before utilizing the dataset, it is essential to preprocess the images to ensure they are suitable for steganographic embedding. This preprocessing includes:

- **Null Value Removal:** Ensuring that there are no corrupt or incomplete images in the dataset. Any image files with null values or missing data are either corrected or removed from the dataset.
- **Label Encoding:** For any categorical data associated with the images, such as image type or source, label encoding is performed to convert these categories into numerical values. This step is particularly useful if the dataset includes metadata that can influence the embedding process.

Step 3: Existing Least Significant Bit (LSB) Substitution in Single Image Steganography Algorithm

To establish a baseline for performance comparison, we first implement the traditional Least Significant Bit (LSB) substitution technique. In LSB steganography, the least significant bit of each pixel in an image is replaced with the bits of the secret message. This method is straightforward and widely used due to its simplicity and ease of implementation. However, it is also more susceptible to detection and less robust against image manipulations and attacks. By implementing this method, we can evaluate its strengths and weaknesses and set a benchmark for our proposed approach.

Step 4: Proposed Pixel Value Differencing (PVD) in Multiple Image Steganography

The core of our research lies in developing an advanced steganographic technique using Pixel Value Differencing (PVD) across multiple images. This involves several key steps:

- **Message Slicing:** The secret message is divided into smaller chunks, each of which will be embedded into different images. This distribution enhances security, as detecting and extracting the entire message requires access to all the carrier images.
- **PVD Embedding:** For each image, the PVD algorithm is applied. This method takes advantage of the differences in pixel values to embed information. By analyzing the differences between adjacent pixel values, the algorithm can embed bits of the secret message in a way that is less noticeable compared to altering individual pixel values directly.
- **Encoding and Storage:** The images with embedded data are saved, ensuring that the embedded information remains imperceptible to the naked eye and resilient against common image processing operations.

Step 5: Performance Comparison

To evaluate the effectiveness of our proposed PVD-based multi-image steganography technique, we perform a comprehensive performance comparison with the LSB substitution method. This comparison involves:

- **Imperceptibility:** Assessing the visual quality of the steganographic images to ensure that the embedded information is not noticeable. This is typically measured using metrics like Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM).
- **Robustness:** Testing the resilience of the embedded data against various attacks and manipulations, such as compression, resizing, and noise addition.
- **Capacity:** Comparing the amount of data that can be embedded without degrading the image quality.

Step 6: Prediction of Output from Test Data with Pixel Value Differencing (PVD) in Multiple Image Steganography Trained Model Algorithm

Finally, we develop and test a predictive model based on the PVD steganography technique. This involves:

- **Training the Model:** Using a portion of the preprocessed dataset, we train a model to embed and extract data using the PVD technique. The training process fine-tunes the algorithm parameters to optimize the embedding and extraction processes.

- **Testing and Validation:** The trained model is then tested on a separate set of images to validate its performance. We evaluate its accuracy in correctly embedding and extracting the secret message, as well as its robustness against various image alterations.
- **Analysis of Results:** The results from the test data are analyzed to assess the model's effectiveness. This includes evaluating the imperceptibility, robustness, and capacity of the steganographic method.

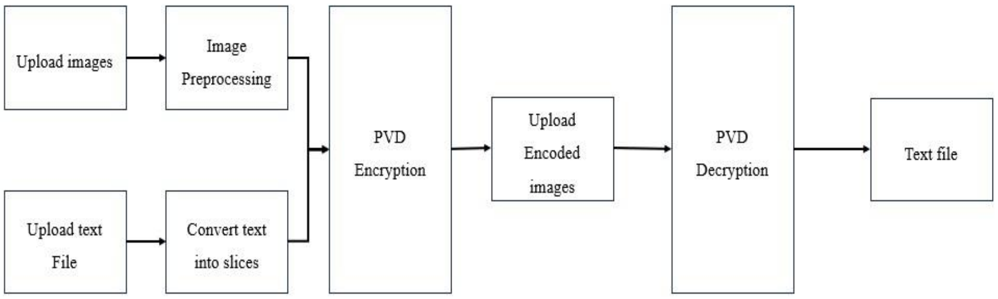


Figure 4.1: Architecture diagram of multiple image steganography

4.2 Pixel Value Differencing (PVD)

Pixel Value Differencing (PVD) is an innovative approach to steganography that operates in the spatial domain of images. While PVD is commonly applied to single images, its extension to multiple image steganography introduces new dimensions of security and capacity. The fundamental concept of PVD revolves around manipulating the pixel values of multiple images to embed hidden information.

In PVD, the difference between the pixel values of adjacent pixels is utilized for data embedding. By carefully adjusting these differences, information can be hidden without significantly altering the visual appearance of the images. PVD operates in the spatial domain, making it resilient to frequency-based attacks. Unlike frequency domain techniques that might be susceptible to transforms, PVD directly modifies pixel values for data concealment.

Multiple Image Steganography

The extension of PVD to multiple images involves distributing hidden information across a set of images. This approach enhances security by dispersing the embedded data, making it more challenging for adversaries to detect or extract the complete message.

Embedding Process:

The embedding process in PVD-based multiple image steganography follows a series of steps:

- Image Preparation: Select a set of cover images for embedding. These images act as carriers for different segments of the hidden message.
- Data Slicing: Divide the hidden message into segments corresponding to the number of selected cover images. Each segment is then embedded into the respective image.
- PVD Embedding: Apply the PVD algorithm to each image, adjusting pixel values based on the differences between adjacent pixels. The differences are manipulated to represent the hidden message.
- Secure Key Integration: Integrate a secure key into the embedding process to enhance security. The key determines how the pixel value differences are adjusted, and without it, extracting the hidden information becomes extremely challenging.

Extraction Process:

The extraction process is designed to retrieve the hidden message from the stego images:

- Image Selection: Choose the stego images containing segments of the hidden message.
- PVD Extraction: Apply the PVD algorithm in reverse to extract the differences between pixel values.
- Data Reconstruction: Reconstruct the hidden message segments from the extracted differences.
- Secure Key Utilization: Use the secure key during the extraction process to ensure accurate retrieval of the hidden information.

APPLICATIONS

- Secure Communication:
  PVD is commonly used for secure communication where confidentiality is paramount. By hiding sensitive information within images, users can communicate covertly without drawing attention to the existence of hidden messages.
- Digital Watermarking:
  PVD finds applications in digital watermarking, where information is embedded within images to assert copyright, ownership, or authenticity. Watermarks created using PVD are often imperceptible, ensuring that they do not degrade the visual quality of the image.
- Covert Data Storage:
  PVD can be utilized for covert data storage within images. Instead of relying on traditional storage methods, data can be hidden within the pixel values of images, providing a stealthy approach to storing and transporting information.
- Image Authentication:
  PVD contributes to image authentication by embedding information that verifies the authenticity and integrity of an image. This is particularly useful in forensic applications and digital forensics.

— Biometric Data Hiding:

PVD can be applied to hide biometric data within images, providing a secure and inconspicuous means of transmitting or storing sensitive biometric information.

— Secure Image Sharing:

PVD enables secure image sharing by allowing users to embed additional information, such as captions or annotations, without visibly altering the image. This is useful in applications where additional data needs to be conveyed alongside the visual content.

— Copyright Protection:

In the realm of digital media, PVD assists in copyright protection by embedding information that identifies the rightful owner or copyright holder. This helps in combating unauthorized use or distribution of digital assets.

— Cryptography Key Exchange:

PVD can be employed for secure key exchange in cryptographic applications. By hiding cryptographic keys within images, users can transmit sensitive information without exposing the keys to potential interception.

— Steganographic Challenges:

PVD is often used in steganographic challenges or competitions where participants are tasked with concealing and extracting hidden messages within images. Such challenges help in testing and improving the robustness of steganographic techniques.

— Academic Research and Experimentation:

PVD serves as a subject of academic research and experimentation in the field of steganography. Researchers explore variations, improvements, and novel applications of PVD to advance the state of the art in information hiding.

— Medical Imaging Security:

In medical imaging, PVD can be applied to embed patient information or metadata within images securely. This ensures that relevant details are associated with medical images without compromising patient privacy.

— Secure Metadata Embedding:

PVD can be used to embed metadata within images, providing a secure way to associate additional information, such as timestamps, geolocation, or camera settings, without altering the visual content significantly.

4.3 Advantages

- Enhanced Security: Distributing hidden information across multiple images improves security by fragmenting the data. Adversaries need access to all images and knowledge of the embedding process to retrieve the complete message.
- Spatial Domain Robustness: PVD operates in the spatial domain, making it robust against certain frequency-based attacks. It is less susceptible to transformations that might affect frequency components.
- Capacity and Redundancy: Multiple images provide a higher capacity for data hiding compared to single-image techniques. Redundancy across images can also contribute to error detection and correction.
- Adaptability: PVD in multiple image steganography is adaptable to various types of images and can be adjusted based on the requirements of the application.

CHAPTER 5

UML DAIGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

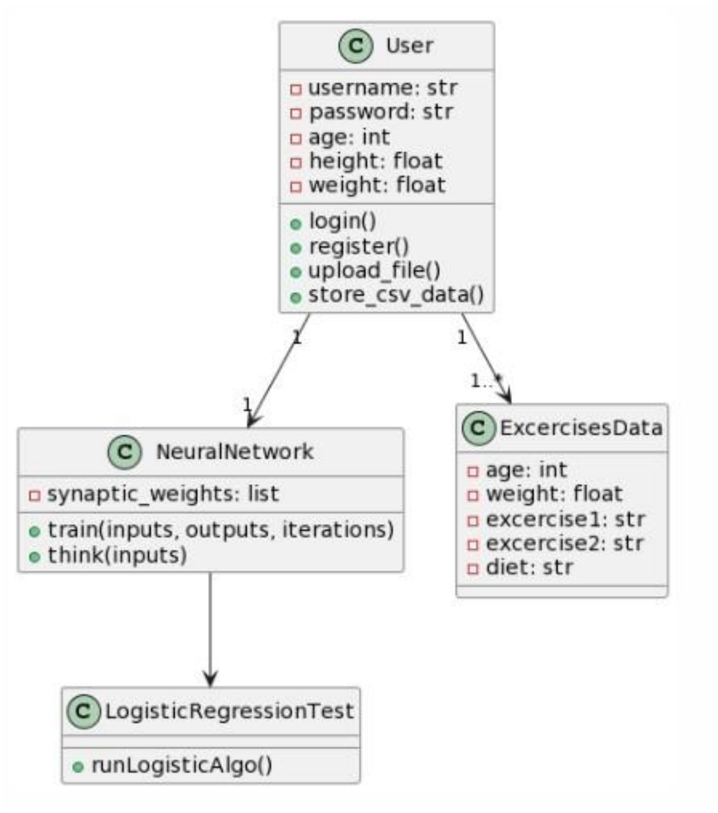
The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS: The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development process.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of OO tools market.
- Support higher level development concepts such as collaborations, frameworks, patterns and components.
- Integrate best practices.

Class diagram

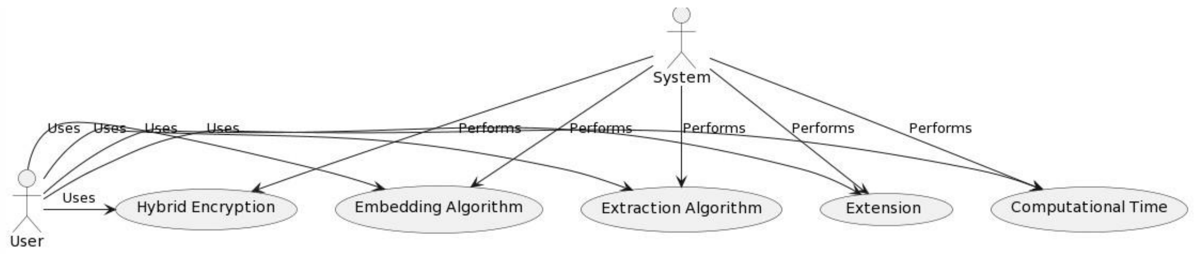
The class diagram is used to refine the use case diagram and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an "is-a" or "has-a" relationship. Each class in the class diagram may be capable of providing certain functionalities. These functionalities provided by the class are termed "methods" of the class. Apart from this, each class may have certain "attributes" that uniquely identify the class.



Use case Diagram

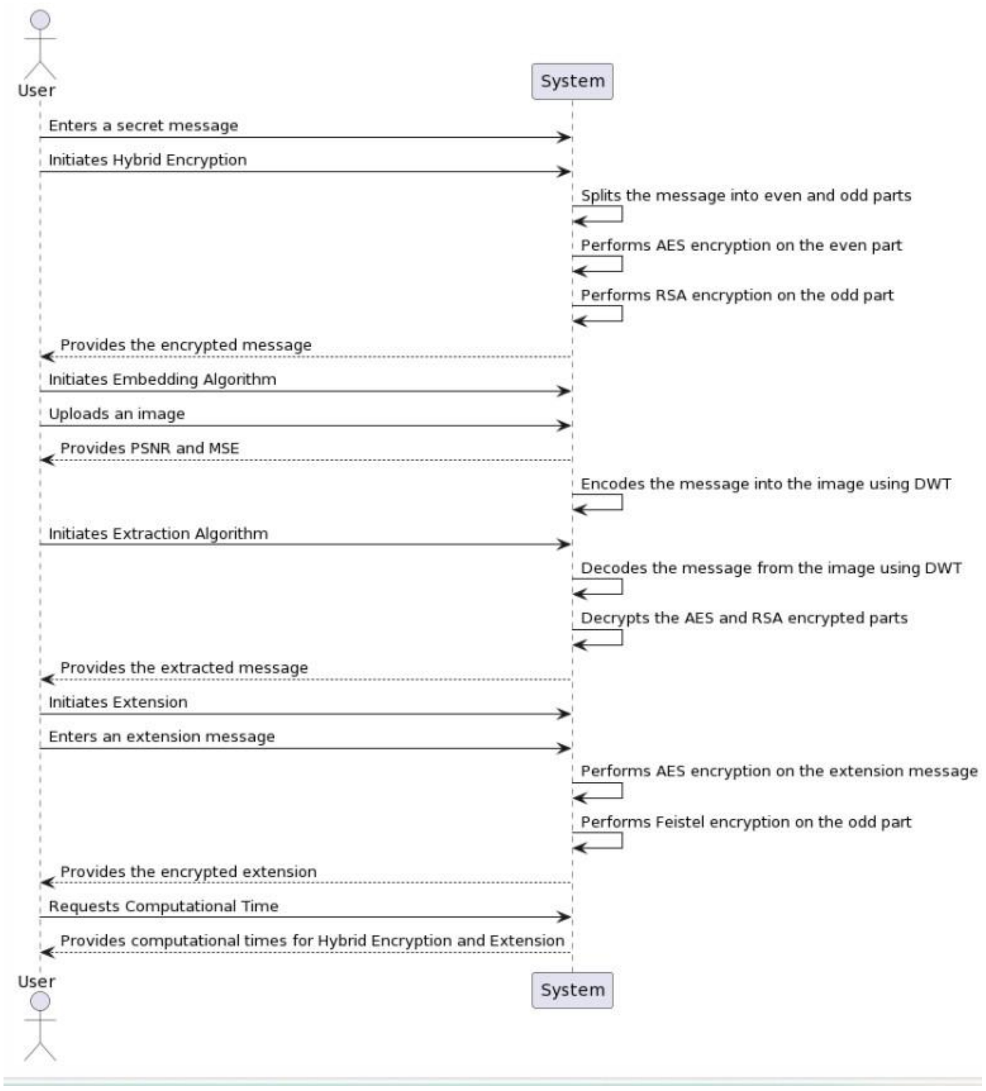
A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor.

Roles of the actors in the system can be depicted.



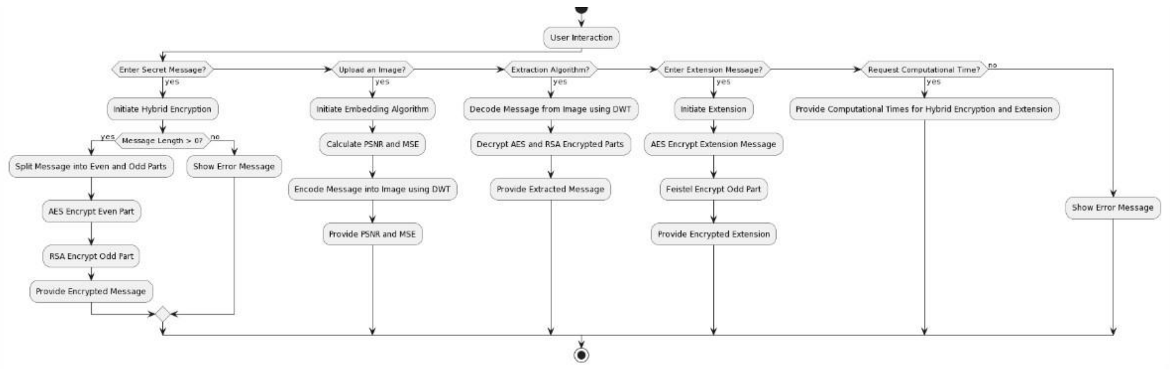
Sequence Diagram

A **sequence diagram** in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows, as parallel vertical lines ("lifelines"), different processes or objects that live simultaneously, and as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.


User
System
Enters a secret message
Initiates Hybrid Encryption
Splits the message into even and odd parts
Performs AES encryption on the even part
Performs RSA encryption on the odd part
Provides the encrypted message
Initiates Embedding Algorithm
Uploads an image
Provides PSNR and MSE
Encodes the message into the image using DWT
Initiates Extraction Algorithm
Decodes the message from the image using DWT
Decrypts the AES and RSA encrypted parts
Provides the extracted message
Initiates Extension
Enters an extension message
Performs AES encryption on the extension message
Performs Feistel encryption on the odd part
Provides the encrypted extension
Requests Computational Time
Provides computational times for Hybrid Encryption and Extension
User
System

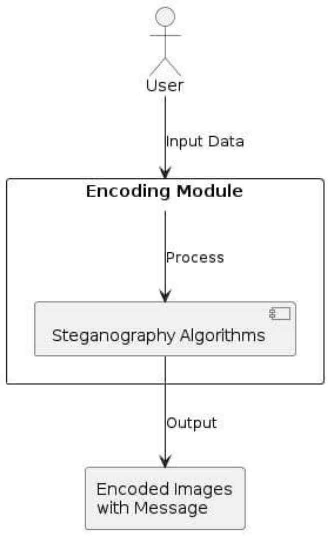
Activity diagram:

Activity diagrams are graphical representations of Workflows of stepwise activities and actions with support for choice, iteration, and concurrency.In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

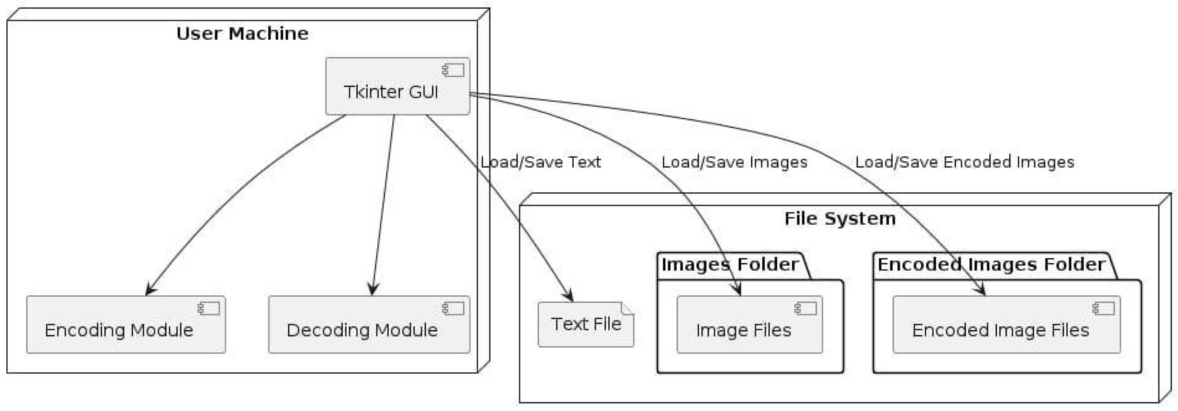

User Interaction
Enter Secret Message?
yes
Initiate Hybrid Encryption
yes
Message Length > 0?
no
Split Message into Even and Odd Parts
Show Error Message
AES Encrypt Even Part
RSA Encrypt Odd Part
Provide Encrypted Message
Upload an Image?
yes
Initiate Embedding Algorithm
Calculate PSNR and MSE
Encode Message into Image using DWT
Provide PSNR and MSE
Extraction Algorithm?
yes
Decode Message from Image using DWT
Decrypt AES and RSA Encrypted Parts
Provide Extracted Message
Enter Extension Message?
yes
Initiate Extension
AES Encrypt Extension Message
Feistel Encrypt Odd Part
Provide Encrypted Extension
Request Computational Time?
yes
no
Provide Computational Times for Hybrid Encryption and Extension
Show Error Message

DataFlow Diagram:

A Data Flow Diagram (DFD) is a visual representation of the flow of data within a system or process. It is a structured technique that focuses on how data moves through different processes and data stores within an organization or a system. DFDs are commonly used in system analysis and design to understand, document, and communicate data flow and processing.



Deployement Diagram: The deployment diagram visualizes the physical hardware on which the software will be deployed



CHAPTER 6

SOFTWARE ENVIRONMENT

What is Python?

Below are some facts about Python.

- Python is currently the most widely used multi-purpose, high-level programming language.
- Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.
- Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.
- Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is huge collection of standard libraries which can be used for the following –

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like Opencv, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia

Advantages of Python

Let's see how Python dominates over other languages.

1. Extensive Libraries

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don't have to write the complete code for that manually.

2. Extensible

As we have seen earlier, Python can be extended to other languages. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add scripting capabilities to our code in the other language.

4. Improved Productivity

The language's simplicity and extensive libraries render programmers more productive than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet of Things. This is a way to connect the language with the real world.

6. Simple and Easy

When working with Java, you may have to create a class to print 'Hello World'. But in Python, just a print statement will do. It is also quite easy to learn, understand, and code. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and indentation is mandatory. These further aids the readability of the code

8. Object-Oriented

This language supports both the procedural and object-oriented programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the encapsulation of data and functions into one.

9. Free and Open-Source

Like we said earlier, Python is freely available. But not only can you download Python for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to code only once, and you can run it anywhere. This is called Write Once Run Anywhere (WORA). However, you need to be careful enough not to include any system-dependent features.

11. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, debugging is easier than in compiled languages.

Any doubts till now in the advantages of Python? Mention in the comment section.

Advantages of Python Over Other Languages

1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and machine learning, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

Disadvantages of Python

So far, we've seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let's now see the downsides of choosing Python over another language.

1. Speed Limitations

We have seen that Python code is executed line by line. But since Python is interpreted, it often results in slow execution. This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the client-side. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called Carbonnelle.

The reason it is not so famous despite the existence of Brython is that it isn’t that secure.

3. Design Restrictions

As you know, Python is dynamically-typed. This means that you don’t need to declare the type of variable while writing the code. It uses duck-typing. But wait, what’s that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can raise run-time errors.

4. Underdeveloped Database Access Layers

Compared to more widely used technologies like JDBC (Java DataBase Connectivity) and ODBC (Open DataBase Connectivity), Python’s database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

5. Simple

No, we’re not kidding. Python’s simplicity can indeed be a problem. Take my example. I don’t do Java, I’m more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

History of Python

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde &Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners[1], Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it. "Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So, I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers." **Python Development Steps**

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991.

This release included already exception handling, functions, and the core data types of lists, dict, str and others. It was also object oriented and had a module system.

Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked. Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting unicode. Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it."Some changes in Python 7.3:

- Print is now a function.
- Views and iterators instead of lists
- The rules for ordering comparisons have been simplified. E.g., a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
- There is only one integer type left, i.e., int. long is int as well.
- The division of two integers returns a float instead of an integer. "//" can be used to have the "old" behaviour.
- Text Vs. Data Instead of Unicode Vs. 8-bit

Purpose

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with lowquality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking. **Modules Used in Project**

TensorFlow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

NumPy

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary datatypes can be defined using NumPy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object-oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

Install Python Step-by-Step in Windows and Mac

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

How to Install Python on Windows and Mac

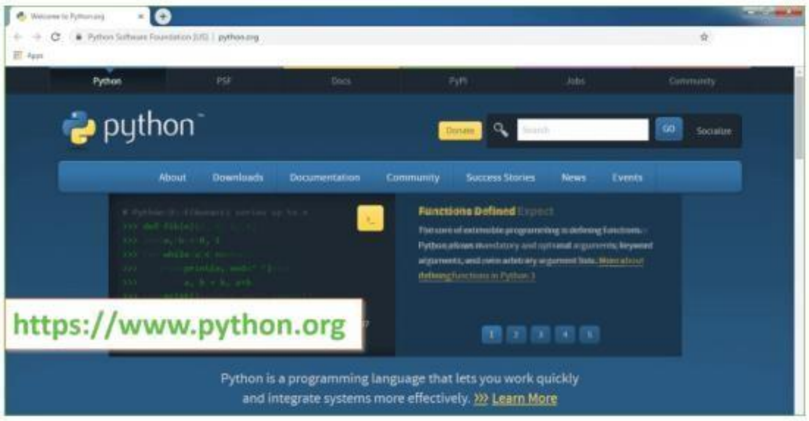
There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your System Requirements. Based on your system type i.e., operating system and based processor, you must download the python version. My system type is a Windows 64-bit operating system. So, the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. Download the Python Cheatsheet here. The steps on how to install Python on Windows 10, 8 and 7 are divided into 4 parts to help understand better.

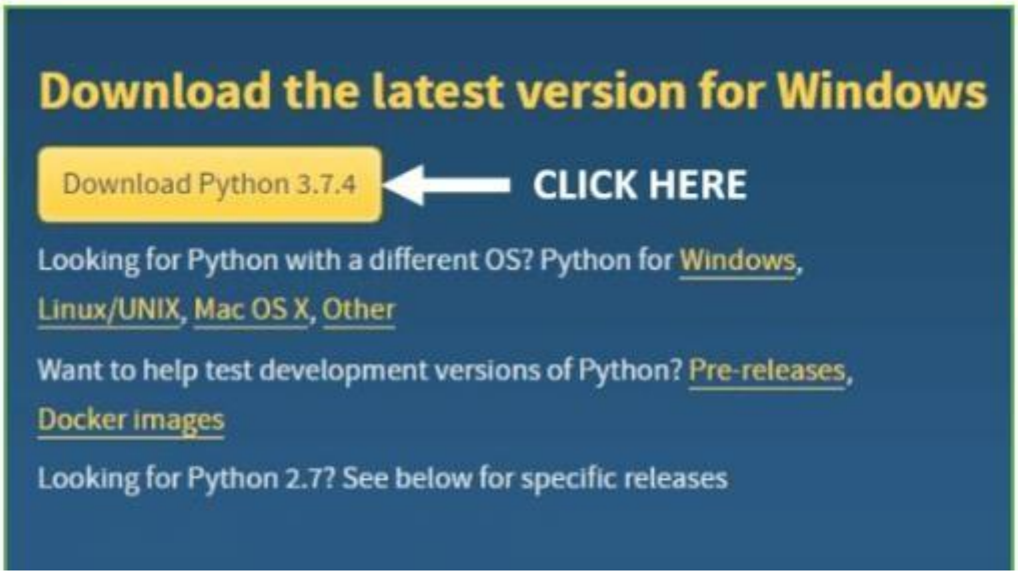
Download the Correct version into the system

Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: https://www.python.org


python
About
Downloads
Documentation
Community
Success Stories
News
Events
https://www.python.org
Python is a programming language that lets you work quickly and integrate systems more effectively. >>> Learn More

Now, check for the latest and the correct version for your operating system.

Step 2: Click on the Download Tab.

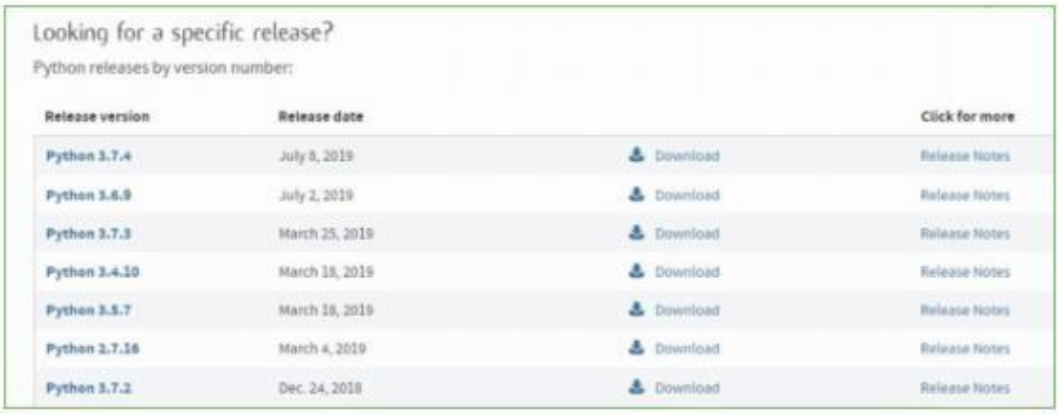


Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

Want to help test development versions of Python? Pre-releases, Docker images

Looking for Python 2.7? See below for specific releases

Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4



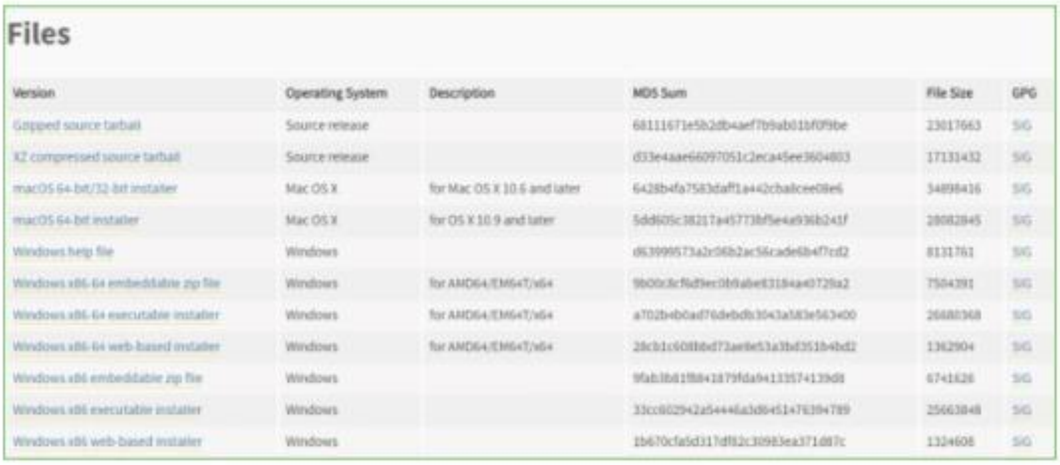
Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.7.4	July 8, 2019	Download	Release Notes
Python 3.6.9	July 2, 2019	Download	Release Notes
Python 3.7.3	March 25, 2019	Download	Release Notes
Python 3.4.10	March 18, 2019	Download	Release Notes
Python 3.5.7	March 18, 2019	Download	Release Notes
Python 2.7.16	March 4, 2019	Download	Release Notes
Python 3.7.2	Dec. 24, 2018	Download	Release Notes

Step 4: Scroll down the page until you find the Files option.

Step 5: Here you see a different version of python along with the operating system.



Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		68111671e5b2db4aef7b9ab01bf0f9be	23017663	SIG
XZ compressed source tarball	Source release		d33e4aae66097051c2eca45ee3604803	17131432	SIG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	6428b4fa7583daff1a442cba8cee08e6	34898416	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5dd605c38217a45773bf5e4a936b241f	28082845	SIG
Windows help file	Windows		d63999573a2c06b2ac56cade6b4f7cd2	8131761	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	9b00c8cf6d9ec0b9abe83184a40729a2	7504391	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b4b0ad76debdb3043a583e563400	26680368	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28cb1c608bbd73ae8e53a3bd351b4bd2	1362904	SIG
Windows x86 embeddable zip file	Windows		9fab3b81f8841879fda94133574139d8	6741626	SIG
Windows x86 executable installer	Windows		33cc602942a54446a3d6451476394789	25663848	SIG
Windows x86 web-based installer	Windows		1b670cfa5d317df82c30983ea371d87c	1324608	SIG

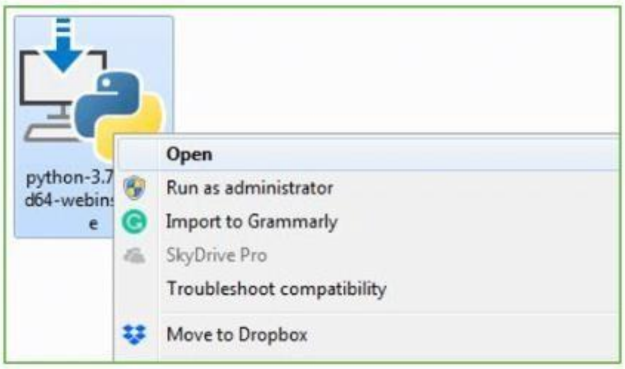
- To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.
- To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer. Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e.,

Installation

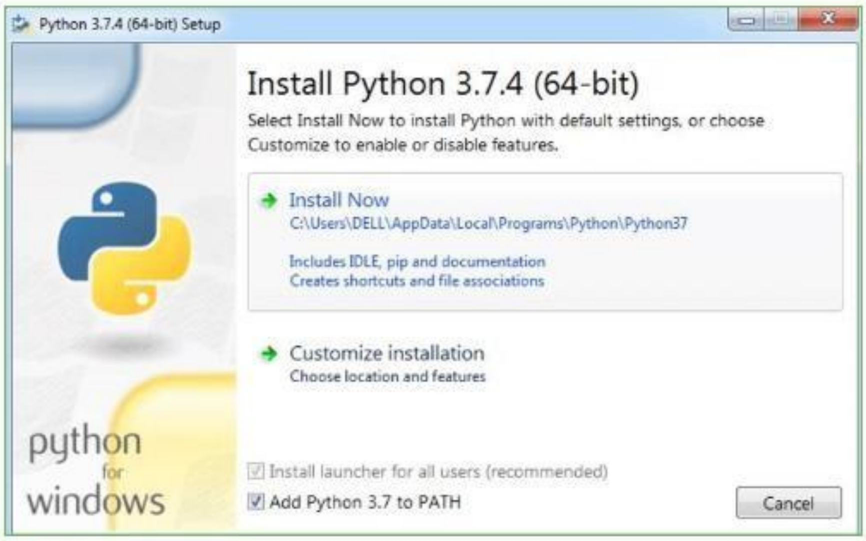
Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python

Step 1: Go to Download and Open the downloaded python version to carry out the installation process.



Step 2: Before you click on Install Now, make sure to put a tick on Add Python 3.7 to PATH.


Python 3.7.4 (64-bit) Setup

Install Python 3.7.4 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

Install Now
C:\Users\DELL\AppData\Local\Programs\Python\Python37

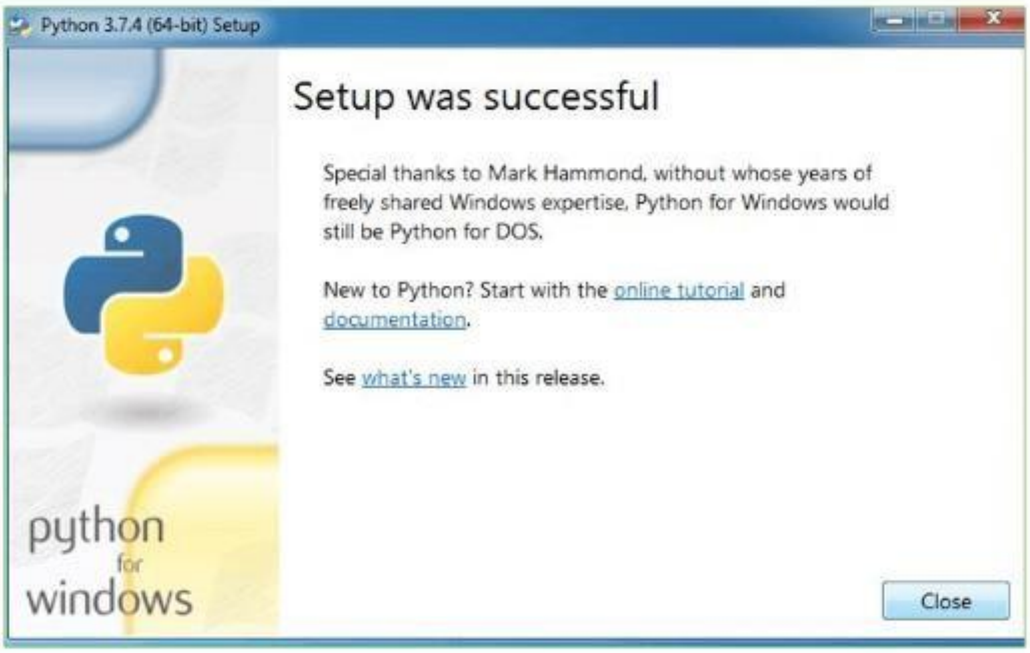
Includes IDLE, pip and documentation
Creates shortcuts and file associations

Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☑ Add Python 3.7 to PATH

Cancel

Step 3: Click on Install NOW After the installation is successful. Click on Close.



Python 3.7.4 (64-bit) Setup

Setup was successful

Special thanks to Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.

New to Python? Start with the online tutorial and documentation.

See what's new in this release.

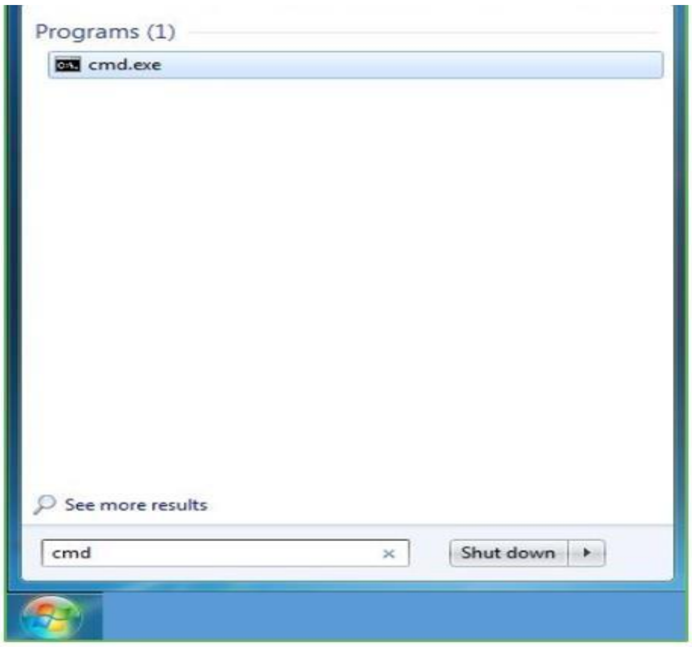
With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

Note: The installation process might take a couple of minutes.

Verify the Python Installation

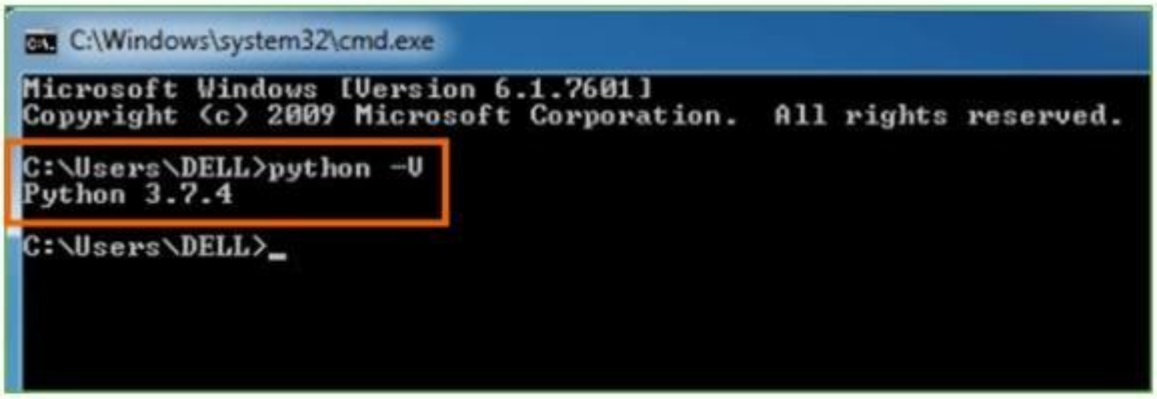
Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.



Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type python –V and press Enter.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\DELL>python -V
Python 3.7.4

C:\Users\DELL>_
```

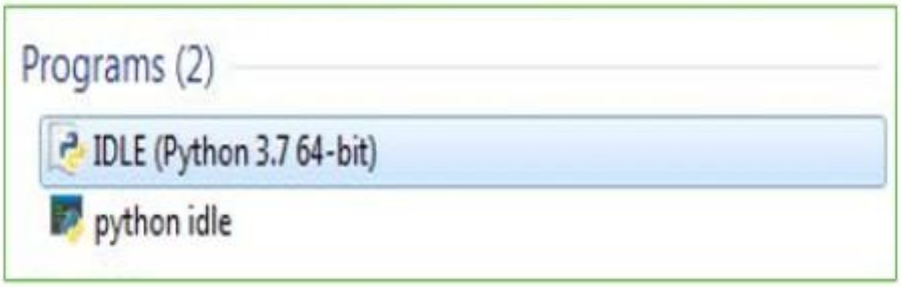
Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE works

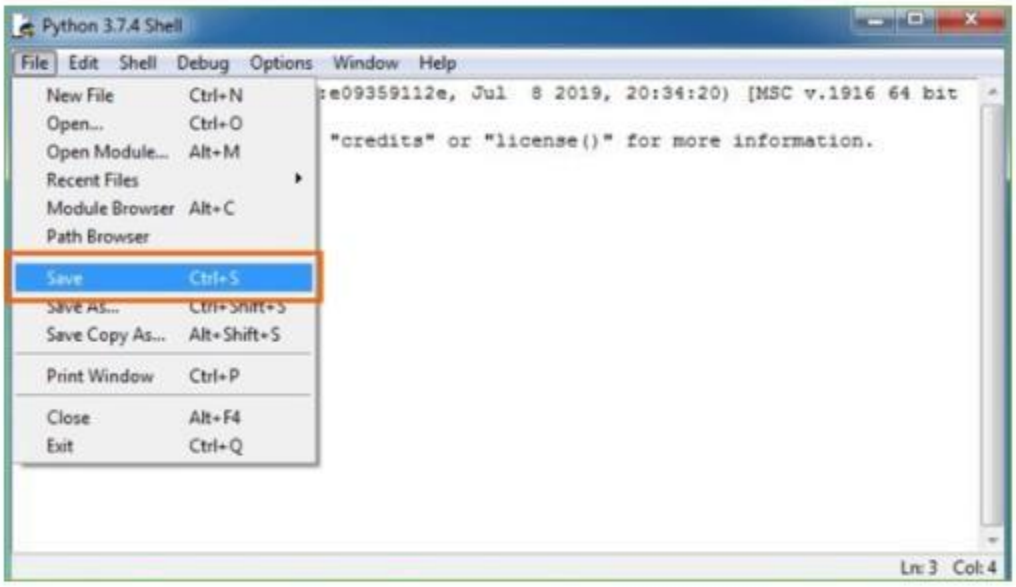
Step 1: Click on Start

Step 2: In the Windows Run command, type “python idle”.



Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

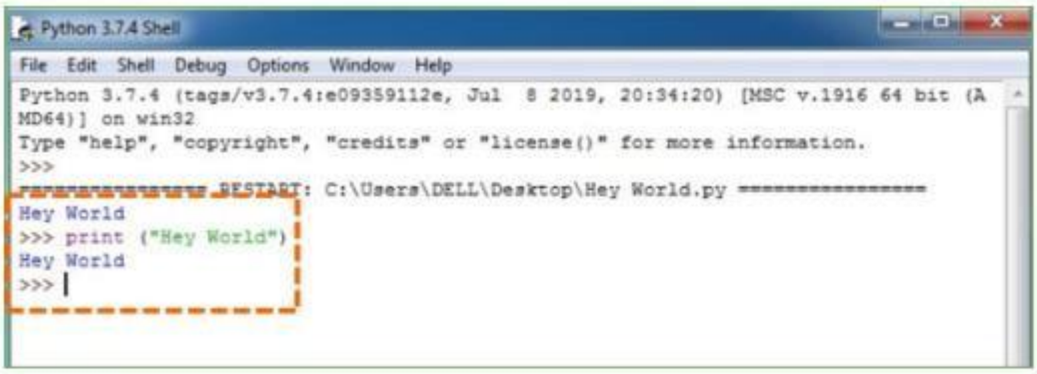
Step 4: To go ahead with working in IDLE you must first save the file. Click on File > Click on Save



```
:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit
 "credits" or "license()" for more information.
```

Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g., enter print ("Hey World") and Press Enter.



```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit (A
MD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
=============== RESTART: C:\Users\DELL\Desktop\Hey World.py ===============
Hey World
>>> print ("Hey World")
Hey World
>>> 
```

You will see that the command given is launched. With this, we end our tutorial on how to install Python. You have learned how to download python for windows into your respective operating system.

Note: Unlike Java, Python does not need semicolons at the end of the statements otherwise it won't work.

CHAPTER 7

SYSTEM REQUIREMENTS SPECIFICATIONS

Software Requirements

The functional requirements or the overall description documents include the product perspective and features, operating system and operating environment, graphics requirements, design constraints and user documentation.

The appropriation of requirements and implementation constraints gives the general overview of the project in regard to what the areas of strength and deficit are and how to tackle them.

- Python IDLE 3.7 version (or)
- Anaconda 3.7 (or)
- Jupiter (or)
- Google colab

Hardware Requirements

Minimum hardware requirements are very dependent on the particular software being developed by a given Enthought Python / Canopy / VS Code user. Applications that need to store large arrays/objects in memory will require more RAM, whereas applications that need to perform numerous calculations or tasks more quickly will require a faster processor.

Operating system	:	Windows, Linux
Processor	:	minimum intel i3
Ram	:	minimum 4 GB
Hard disk	:	minimum 250GB

CHAPTER 8

FUNCTIONAL REQUIREMENTS

Output Design

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provides a permanent copy of the results for later consultation. The various types of outputs in general are:

- External Outputs, whose destination is outside the organization
- Internal Outputs whose destination is within organization and they are the
- User's main interface with the computer.
- Operational outputs whose use is purely within the computer department.
- Interface outputs, which involve the user in communicating directly.

Output Definition

The outputs should be defined in terms of the following points:

- Type of the output
- Content of the output
- Format of the output
- Location of the output
- Frequency of the output
- Volume of the output
- Sequence of the output

It is not always desirable to print or display data as it is held on a computer. It should be decided as which form of the output is the most suitable. **Input Design**

Input design is a part of overall system design. The main objective during the input design is as given below:

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that the input is acceptable and understood by the user. **Input Stages**

The main input stages can be listed as below:

- Data recording
- Data transcription
- Data conversion
- Data verification

- Data control
- Data transmission
- Data validation
- Data correction **Input Types**

It is necessary to determine the various types of inputs. Inputs can be categorized as follows:

- External inputs, which are prime inputs for the system.
- Internal inputs, which are user communications with the system.
- Operational, which are computer department's communications to the system?
- Interactive, which are inputs entered during a dialogue.

Input Media

At this stage choice has to be made about the input media. To conclude about the input media consideration has to be given to;

- Type of input
- Flexibility of format
- Speed
- Accuracy
- Verification methods
- Rejection rates
- Ease of correction
- Storage and handling requirements
- Security
- Easy to use
- Portability

Keeping in view the above description of the input types and input media, it can be said that most of the inputs are of the form of internal and interactive. As

Input data is to be the directly keyed in by the user, the keyboard can be considered to be the most suitable input device.

Error Avoidance

At this stage care is to be taken to ensure that input data remains accurate form the stage at which it is recorded up to the stage in which the data is accepted by the system. This can be achieved only by means of careful control each time the data is handled.

Error Detection

Even though every effort is made to avoid the occurrence of errors, still a small proportion of errors is always likely to occur, these types of errors can be discovered by using validations to check the input data.

Data Validation

Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area where there is a possibility for the user to commit errors. The system will not accept invalid data. Whenever an invalid data is keyed in, the system immediately prompts the user and the user has to again key in the data and the system will accept the data only if the data is correct. Validations have been included where necessary.

The system is designed to be a user friendly one. In other words the system has been designed to communicate effectively with the user. The system has been designed with popup menus.

User Interface Design

It is essential to consult the system users and discuss their needs while designing the user interface:

User Interface Systems Can Be Broadly Clasified As:

- User initiated interface the user is in charge, controlling the progress of the user/computer dialogue. In the computer-initiated interface, the computer selects the next stage in the interaction.
- Computer initiated interfaces

In the computer-initiated interfaces the computer guides the progress of the user/computer dialogue.
Information is displayed and the user response of the computer takes action or displays further information.

User Initiated Intergfaces

User initiated interfaces fall into two approximate classes:

- Command driven interfaces: In this type of interface the user inputs commands or queries which are interpreted by the computer.
- Forms oriented interface: The user calls up an image of the form to his/her screen and fills in the form. The forms-oriented interface is chosen because it is the best choice. **Computer-**

Initiated Interfaces

The following computer – initiated interfaces were used:

- The menu system for the user is presented with a list of alternatives and the user chooses one; of alternatives.
- Questions – answer type dialog system where the computer asks question and takes action based on the basis of the users reply.

Right from the start the system is going to be menu driven, the opening menu displays the available options. Choosing one option gives another popup menu with more options. In this way every option leads the users to data entry form where the user can key in the data.

Error Message Design

The design of error messages is an important part of the user interface design. As user is bound to commit some errors or other while designing a system the system should be designed to be helpful by providing the user with information regarding the error he/she has committed.

This application must be able to produce output at different modules for different inputs.

Performance Requirements

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely in the part of the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

- The system should be able to interface with the existing system
- The system should be accurate
- The system should be better than the existing system
- The existing system is completely dependent on the user to perform all the duties.

CHAPTER 9

SOURCE CODE

```
from tkinter import messagebox from
tkinter import * from tkinter import
simpledialog import tkinter from tkinter
import filedialog from tkinter.filedialog
import askopenfilename import numpy as
np import cv2 import os from pvd_lib
import pvd_lib

main = Tk() main.title("Advancing Image Steganography And
Classification ") main.geometry("1300x1200")

global image_path, text_path

pvd_obj = pvd_lib()

def  sliceText(text_file,  images_path):
with open(text_file, "rb") as myfile:
data = myfile.read() myfile.close() files = []
for root, dirs, directory in
os.walk(images_path): for j in
range(len(directory)): name =
os.path.basename(root) if 'Thumbs.db' not in
directory[j]:
files.append(root+"/"+directory[j])

length = len(data) size =
int(length / len(files)) tot_blocks
= len(files) start = 0 end = size
block = [] for i in range(0,
tot_blocks): chunk =
data[start:end]
block.append(chunk.decode('lati
n1')) start = end end = end + size
remain =  length - start if remain
> 0:
chunk = data[start:length]
last = block[len(block)-1]
last +=
chunk.decode('latin1')
block[len(block)-1] = last
print(str(start)+"    "+str(length)+"    "+str(chunk.decode('latin1')))
return block, files

def PVDEncoding(image_path, slice_msg):
base_name =
os.path.basename(os.path.dirname(image_path)) with
open("data.txt", "wb") as myfile:
myfile.write(slice_msg.encode()) myfile.close() if
os.path.exists('Encoded_Images/'+base_name) == False:
os.makedirs('Encoded_Images/'+base_name) img_name =
os.path.basename(image_path) img_name =
img_name.replace(".jpg", ".png")
pvd_obj.pvd_embed(image_path, "data.txt",
'Encoded_Images/'+base_name+"/"+img_name)

def PVDDecoding(image_path):
output = "" base_name =
os.path.basename(image_path) files = [] for
root, dirs, directory in
os.walk(image_path): for j in
range(len(directory)): name =
os.path.basename(root) if 'Thumbs.db' not
in directory[j]:
files.append(root+"/"+directory[j]) for i in
range(len(files)):
img_name = os.path.basename(files[i]) img_name = img_name.replace(".jpg",
".png") print(files[i]+" ==
"+'Encoded_Images/'+base_name+"/"+os.path.basename(files[i]))
```

```
pvd_obj.pvd_extract(files[i], "data.txt",
'Encoded_Images/'+base_name+"/"+img_name) with open("data.txt", "rb") as
myfile:
data = myfile.read()
myfile.close() output +=
data.decode('latin1') return
output

def uploadImage():
global image_path
text.delete('1.0',
END)
image_path = filedialog.askdirectory(initialdir =
".") text.insert(END,image_path+" loaded\n\n")
tf1.insert(0, image_path)

def uploadText():
global text_path, image_path text_path
= askopenfilename(initialdir = ".")
tf2.insert(0, text_path)
text.insert(END,text_path+" loaded")
block, files = sliceText(text_path, image_path) for
i in range(len(files)): img = cv2.imread(files[i])
img = cv2.resize(img, (600,600)) img =
cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
cv2.imwrite(files[i], img) PVDEncoding(files[i],
block[i]) text.insert(END,"Slice Message :
"+str(block[i])+"\n") text.insert(END,"Hidden
inside : "+str(files[i])+"\n\n")
text.update_idletasks() text.update()

def ExtractText():
text.delete('1.0', END) encoded_path =
filedialog.askdirectory(initialdir = "Encoded_Images") tf3.insert(0,
encoded_path) name = os.path.basename(encoded_path)
```

```
print("extract "+name) output = PVDDecoding(name)
text.insert(END,"Extracted Text = "+output)



font = ('times', 15, 'bold') title = Label(main, text=' Advancing Image Steganography
And Classification ') title.config(bg='mint cream', fg='olive drab')
title.config(font=font) title.config(height=3, width=120) title.place(x=0,y=5) font1 =
('times', 14, 'bold') ff = ('times', 12, 'bold') l1 = Label(main, text='Encoding Images
Folder:') l1.config(font=font1) l1.place(x=50,y=100) tf1 = Entry(main,width=35)
tf1.config(font=font1) tf1.place(x=330,y=100) uploadButton = Button(main,
text="Upload Encoding Images", command=uploadImage)
uploadButton.place(x=720,y=100) uploadButton.config(font=ff) l2 = Label(main,
text='Upload Text File:') l2.config(font=font1) l2.place(x=50,y=150) tf2 =
Entry(main,width=35) tf2.config(font=font1) tf2.place(x=330,y=150)
encodingButton = Button(main, text="Upload Text File to Hide & PVD Encode",
command=uploadText) encodingButton.place(x=720,y=150)
encodingButton.config(font=ff)

l3 = Label(main, text='Upload Folder to Extract
Text:') l3.config(font=font1) l3.place(x=50,y=200)

tf3 = Entry(main,width=35)
tf3.config(font=font1)
tf3.place(x=330,y=200)

encodingButton = Button(main, text="PVD Decoding",
command=ExtractText) encodingButton.place(x=720,y=200)
encodingButton.config(font=ff) font1 = ('times', 13, 'bold')
text=Text(main,height=20,width=120) scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set) text.place(x=10,y=250)
text.config(font=font1) main.config(bg='gainsboro') main.mainloop()
```

CHAPTER 10

RESULTS AND DISCUSSION

10.1 Implementation description

– Tkinter Main Window Setup:

The script initializes the main Tkinter window with a title and dimensions, serving as the GUI interface.

– Global Variables and PVD Object Initialization:

Global variables (image_path and text_path) store paths for the selected image folder and text file. An instance of the pvd_lib class is created to handle PVD operations.

– Text Slicing Function (sliceText):

Reads a binary text file and divides its content into blocks.Iterates through images in a specified folder, returning a list of text blocks and corresponding image file paths.

– PVD Encoding Function (PVDEncoding):

Writes the sliced text message to a temporary file ("data.txt").Creates a folder for encoded images if it doesn't exist. Utilizes the pvd_embed method from the pvd_obj instance to perform PVD encoding on each image.

– PVD Decoding Function (PVDDecoding):

Iterates through encoded images in a specified folder.Uses the pvd_extract method from the pvd_obj instance to perform PVD decoding on each image.Concatenates the extracted data from each image to reconstruct the original hidden text.

– Upload Image Function (uploadImage):

Opens a file dialog allowing the user to select an image folder.Displays the selected image folder path in the GUI.Upload Text Function (uploadText):Opens a file dialog allowing the user to select a text file.Displays the selected text file path in the GUI.Slices the text and performs PVD encoding on each image in the selected image folder.

– Extract Text Function (ExtractText):

Opens a file dialog allowing the user to select a folder containing encoded images.Displays the selected folder path in the GUI. Performs PVD decoding on each encoded image in the selected folder. Concatenates the extracted text from each image to reveal the original hidden message.

– GUI Elements:

Labels, entry widgets, buttons, and a text box create a user-friendly interface. Labels provide information or titles for various sections.Entry widgets allow users to input or display information.Buttons trigger specific actions when clicked.A text box displays information, messages, or the extracted text.

– Tkinter Main Loop:

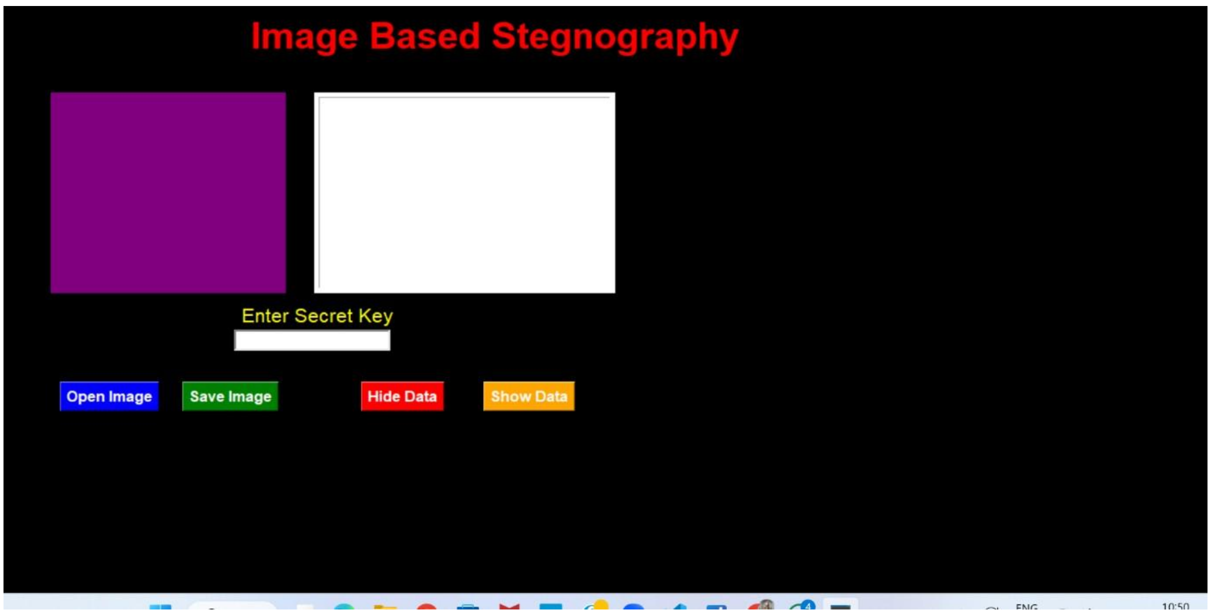
Initiates the Tkinter event loop, allowing the GUI to respond to user interactions and run the application.

10.2 Results and description

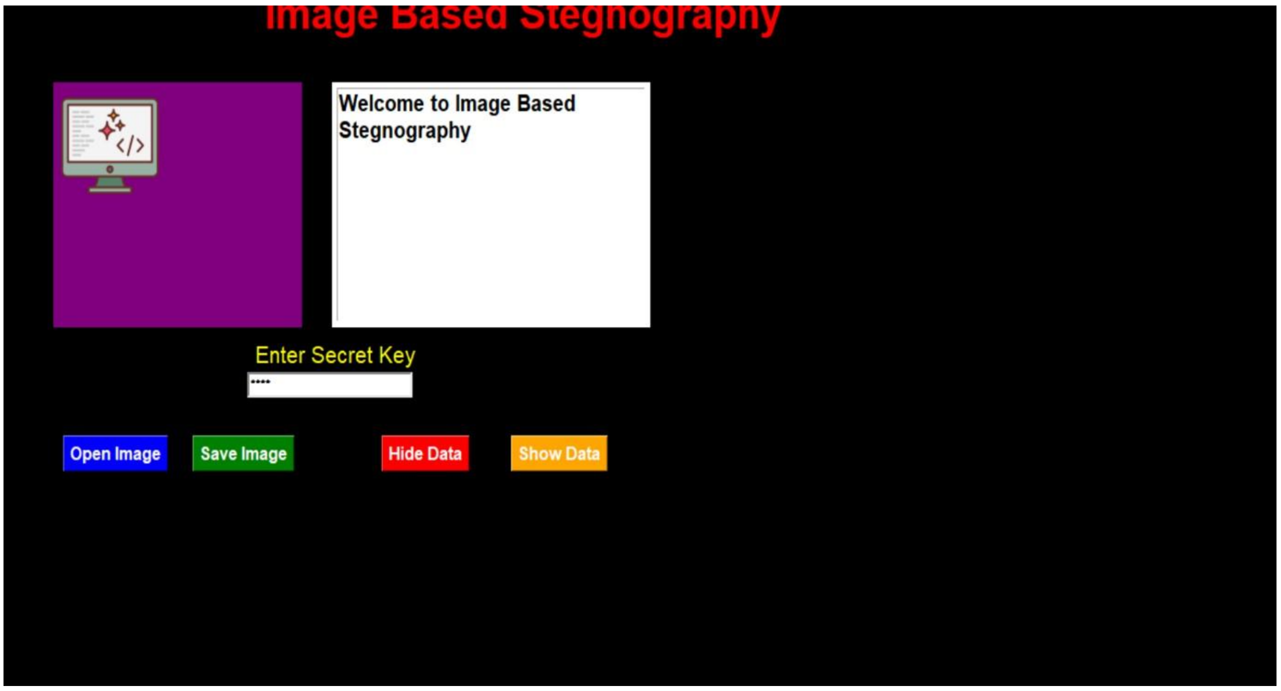
Improved File Security System Using Multiple Image Steganography

In this project as per your instructions we have developed PVD (Pixel Value Differencing) based image steganography where user can upload multiple images folder and then upload text file which has to be slice and embed in all those uploaded images. All embed images will get saved inside 'Encoded_Images' folder with text slice data hidden inside it. While decoding we can upload desired folder from 'Encoded_Images' folder to extract text.

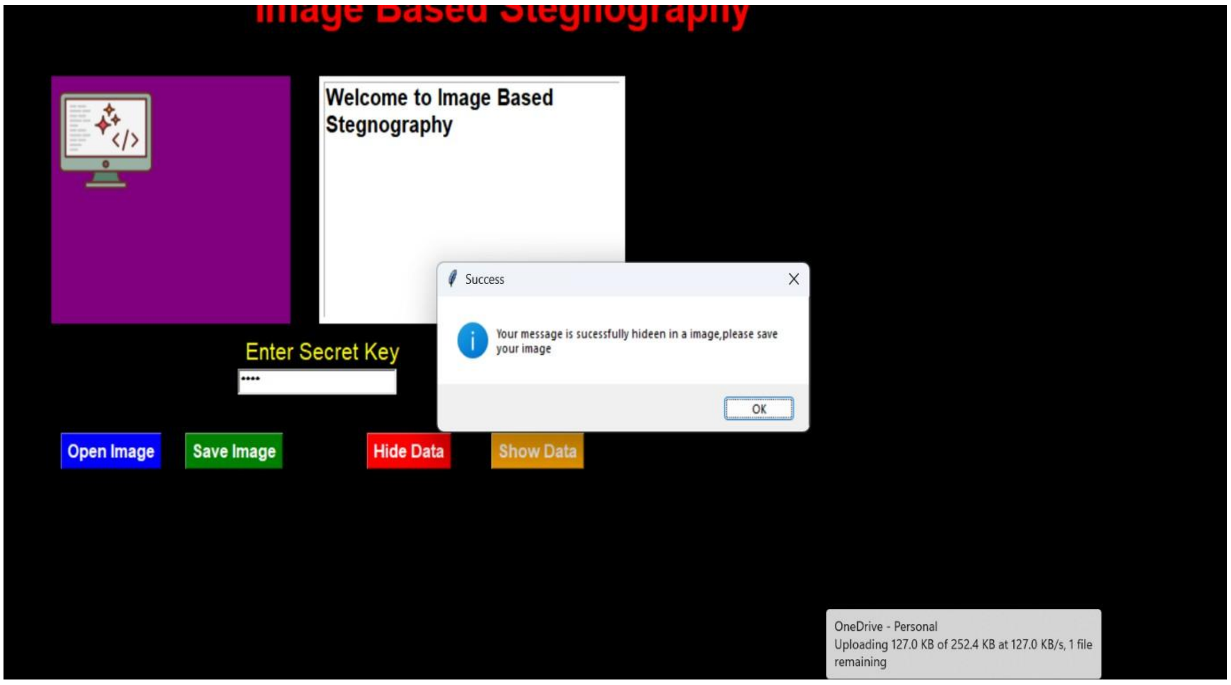
To embed text we are using below sample text file



In the above screen click on open image and upload image



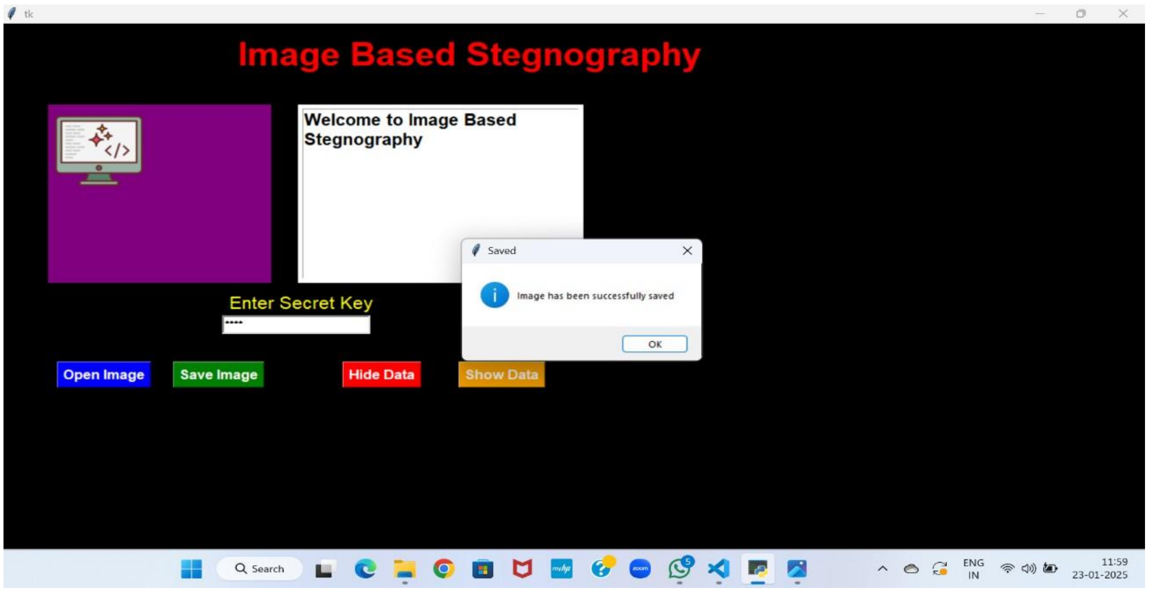
Next enter the secret key and Click on save image



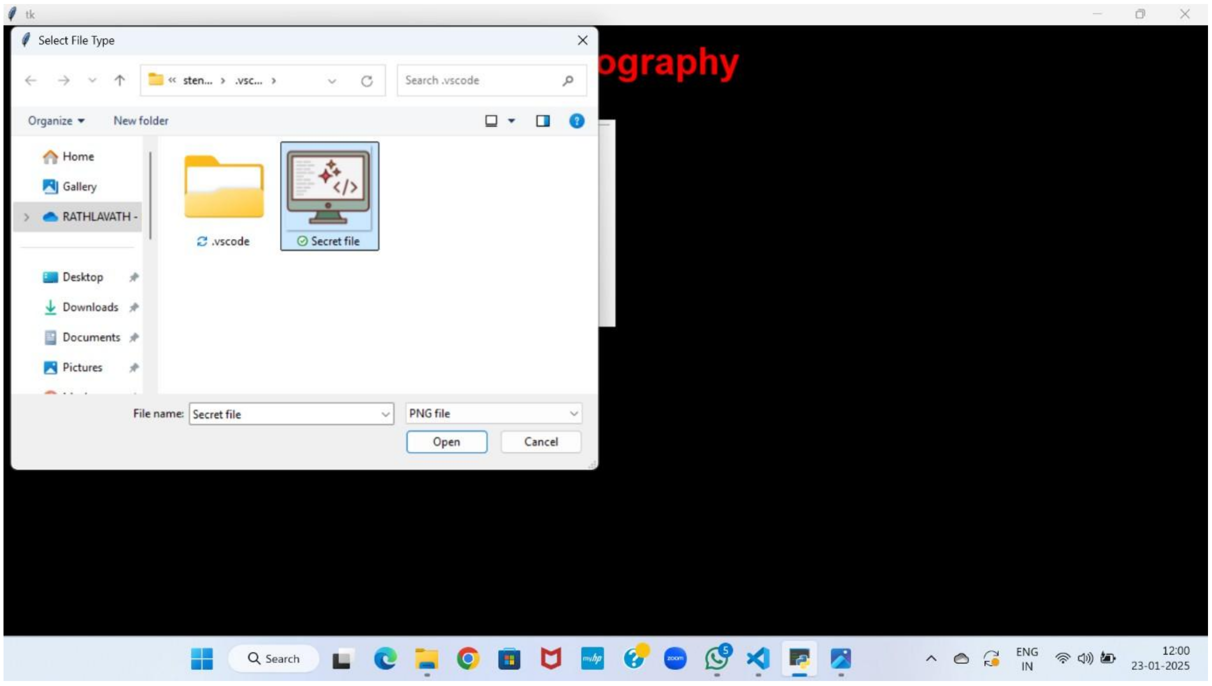
Your message is sucessfully hideen in a image,please save your image

OneDrive - Personal
Uploading 127.0 KB of 252.4 KB at 127.0 KB/s, 1 file remaining

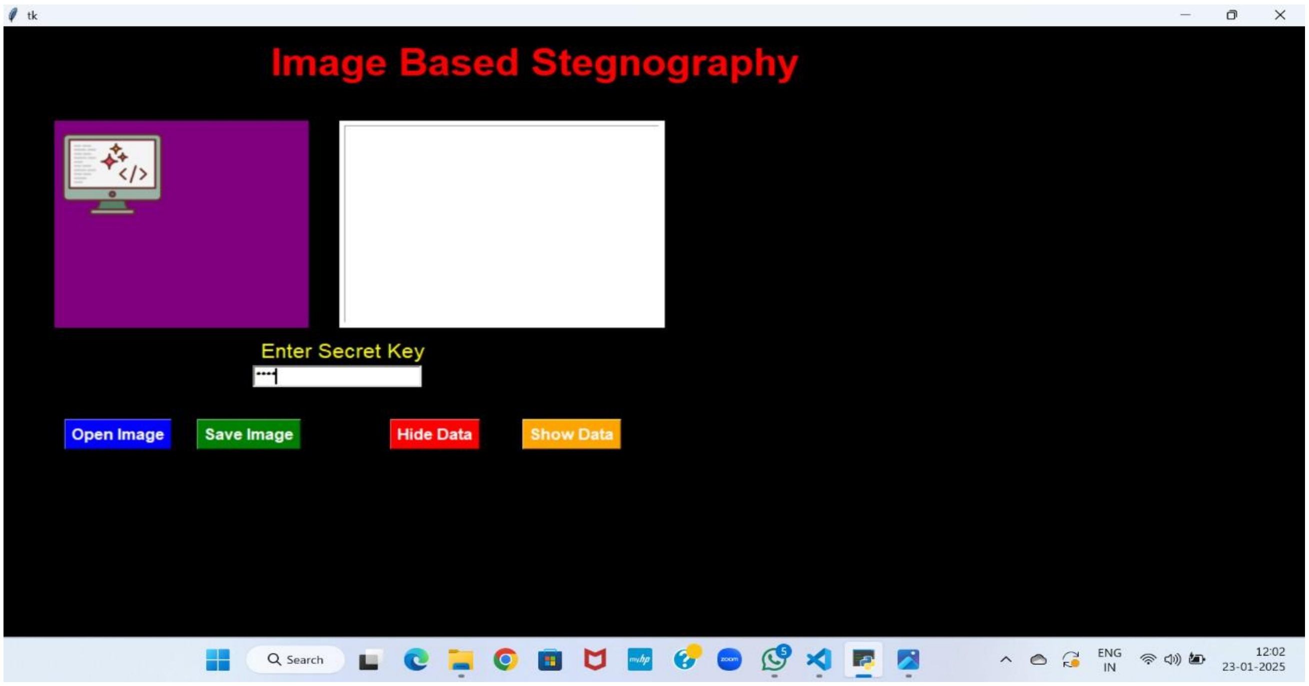
Next click on Hide data the the text is successful hides in a image



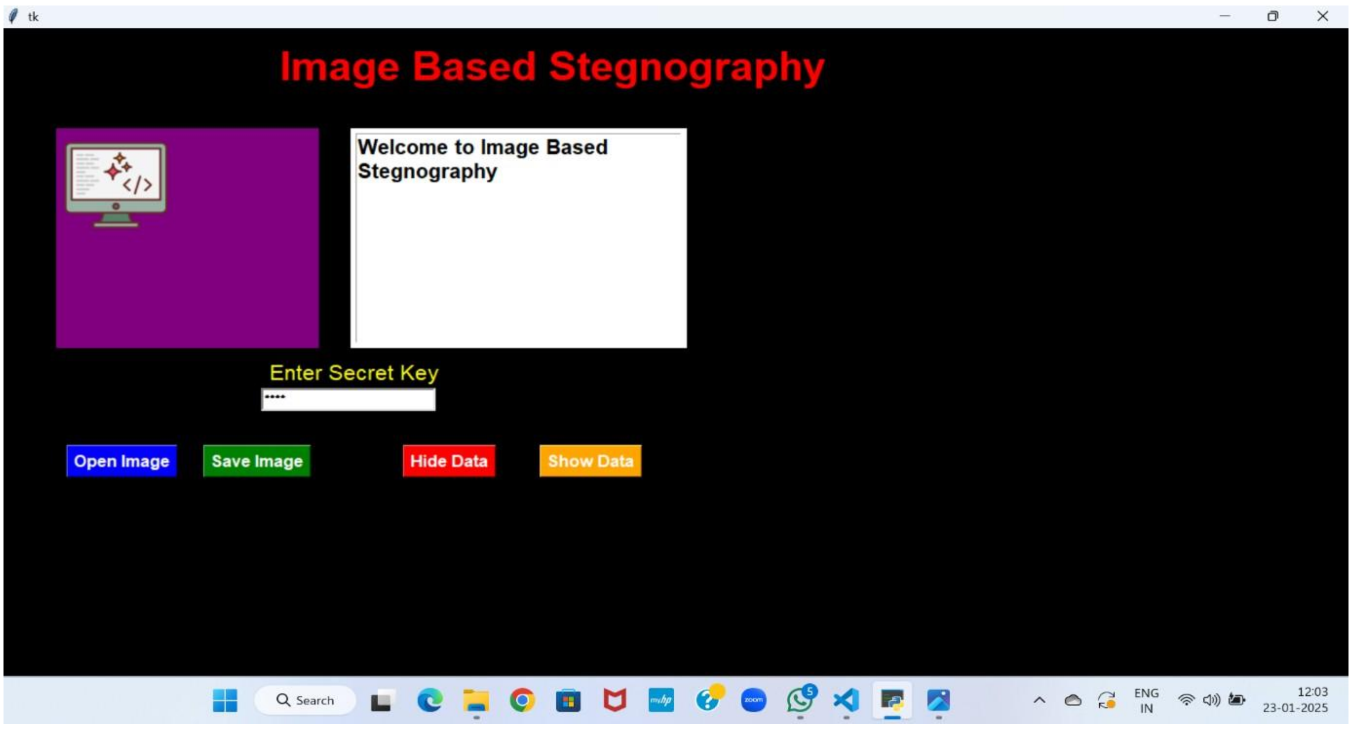
Next click on save image then image is successfully saved



To decrypt text click on open image and upload secret image file



Next Enter your secret key and Click on show data then it shows your decrypted text



Welcome to Image Based Stegnography

CHAPTER 11

CONCLUSION AND FUTURE SCOPE

multiple image steganography represents a significant advancement in the field of covert communication and secure data transmission. The technique of distributing hidden information across a series of images, coupled with the Pixel Value Differencing (PVD) algorithm, offers a potent combination of security, resilience, and imperceptibility. The strategic division of data, error-correction techniques, spread spectrum methods, and secret sharing schemes contribute to the robustness and reliability of the steganographic system.

The incorporation of cryptography and encryption enhances the confidentiality of the concealed information, while authentication and watermarking techniques provide mechanisms for verifying the integrity of the images. Hybrid approaches, integrating various steganographic methods and security measures, offer adaptability and versatility to meet diverse security requirements.

However, the implementation of multiple image steganography is not without challenges. Synchronization during the embedding and extraction processes is crucial for accurate data retrieval. Striking a balance between usability and security is a delicate consideration, requiring thoughtful trade-offs to create an effective and userfriendly steganographic system.

Future Scope

As multiple image steganography continues to evolve, several avenues for future research and development emerge:

Advanced Encryption Techniques: Research into more robust encryption methods can further enhance the security of concealed information, making it even more challenging for adversaries to decipher without the proper key.

Dynamic Data Distribution Strategies: Exploring dynamic strategies for distributing data across images could optimize the security and efficiency of multiple image steganography. Adaptive methods that adjust based on the characteristics of the data and images may yield improved results.

Machine Learning Integration: The integration of machine learning algorithms for both embedding and detection processes could open new possibilities for automating and optimizing multiple image steganography. Machine learning models may adapt to evolving security threats and enhance the overall effectiveness of the technique.

Usability Improvements: Future research should focus on refining the user experience of multiple image steganography tools. Balancing security with user-friendly interfaces and seamless integration into existing workflows is essential for widespread adoption.

Quantum Steganography: Exploring the application of quantum principles to steganography could usher in a new era of secure communication. Quantum steganography may offer unique advantages in terms of security and information concealment.

Real-World Applications: Investigating practical applications of multiple image steganography in diverse domains, such as healthcare, finance, and communication, can uncover novel use cases and contribute to the integration of this technique into real-world scenarios.

REFERENCES

[1]. B. Sultan and M. A. Wani, "Multi-data Image Steganography using Generative Adversarial Networks," 2022 9th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 2022, pp. 454-459, doi: 10.23919/INDIACom54597.2022.9763273.

[2]. X. Liao, J. Yin, M. Chen and Z. Qin, "Adaptive Payload Distribution in Multiple Images Steganography Based on Image Texture Features," in IEEE Transactions on Dependable and Secure Computing, vol. 19, no. 2, pp. 897-911, 1 March-April 2022, doi: 10.1109/TDSC.2020.3004708.

[3]. M. Srivastava, P. Dixit and S. Srivastava, "Data Hiding using Image Steganography," 2023 6th International Conference on Information Systems and Computer Networks (ISCON), Mathura, India, 2023, pp. 1-6, doi:
10.1109/ISCON57294.2023.10112069.

[4]. A. G. Benedict, "Improved File Security System Using Multiple Image Steganography," 2019 International Conference on Data Science and Communication (IconDSC), Bangalore, India, 2019, pp. 1-5, doi: 10.1109/IconDSC.2019.8816946.

[5]. S. Mukhopadhyay and H. Leung, "Multi Image Encryption and Steganography Based on Synchronization of Chaotic Lasers," 2013 IEEE International Conference on Systems, Man, and Cybernetics, Manchester, UK, 2013, pp. 4403-4408, doi: 10.1109/SMC.2013.751.

[6]. A. S. Ansari, M. S. Mohammadi and M. T. Parvez, "A Multiple-Format Steganography Algorithm for Color Images," in IEEE Access, vol. 8, pp. 83926-83939, 2020, doi: 10.1109/ACCESS.2020.2991130.

[7]. P. Grandhe, A. M. Reddy, K. Chillapalli, K. Koppera, M. Thambabathula and L. P. Reddy Surasani, "Improving The Hiding Capacity of Image Steganography with Stego-Analysis," 2023 IEEE International Conference on Integrated Circuits and Communication Systems (ICICACS), Raichur, India, 2023, pp. 0106, doi: 10.1109/ICICACS57338.2023.10100146.

[8]. R. Joshi, A. K. Bairwa, V. Soni and S. Joshi, "Data Security Using Multiple Image Steganography and Hybrid Data Encryption Techniques," 2022 International Conference for Advancement in Technology (ICONAT), Goa, India, 2022, pp. 1-7, doi: 10.1109/ICONAT53423.2022.9725949.

[9]. Z. Wang, Z. Zhang and J. Jiang, "Multi-Feature Fusion based Image Steganography using GAN," 2021 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), Wuhan, China, 2021, pp. 280-281, doi: 10.1109/ISSREW53611.2021.00079.

[10]. X. Zhao and H. Huang, "Research on Image Steganography Based on Multiple Expansion Generation Adversarial Network," 2021 IEEE 3rd International Conference on Frontiers Technology of Information and Computer (ICFTIC), Greenville, SC, USA, 2021, pp. 361-366, doi:
10.1109/ICFTIC54370.2021.9647204.

[11]. B. Wei, X. Duan and H. Nam, "Image Steganography with Deep Learning Networks," 2022 13th International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, Korea, Republic of, 2022, pp. 1371-1374, doi: 10.1109/ICTC55196.2022.9952432.

[12]. M. Liu, W. Luo, P. Zheng and J. Huang, "A New Adversarial Embedding Method for Enhancing Image Steganography," in IEEE Transactions on Information Forensics and Security, vol. 16, pp. 4621-4634, 2021, doi: 10.1109/TIFS.2021.3111748.

Project Details

Academic Year	**2024-2025**
Title of the Project	Hide Secret Text Message Inside Image Using Python | STEGANOGRAPHY |
Name of the Students and Hall Ticket No.	**R. KISHORE (21RA1A0541)** **K. DINESH KUMAR (21RA1A0550)** **V. LAXMAN REDDY (21RA1A0550)**
Name of the Guide	**Mrs. E Sravanthi**

Project PO Mapping

Name of Course From which Principles are applied in This Project	**Related Course Outcomes Number**	**Description of the application**	**Page Number**	**Attained**
Python Programming Software Engineering (C313)	C313.1	Students described the basis for their problem statement.	01	PO2
Machine Learning, Python Programming, Data Mining (C413, C411)	C322.2, C411.2	Students explained about Hide Secret Text Message Inside Image Using Python | STEGANOGRAPHY | Python Programming	1-2	PO1
Software Engineering, Python Programming (C313)	C313.3	Students identified the existing system and its Drawbacks and proposed a Solution to it.	11-12	P02, P03
Software Engineering (C313)	C313.1	Students identified the Hardware and Software required for the project.	40	PO5
Design Patterns, Software Engineering, DBMS (C313, C322)	C313.2, C222.3	Students explain the flow of the project using UML diagrams designed in STAR UML, ER diagram.	21-26	P03, P05, PO9, PSO3

Python Programming	C413.2, C411.2	Students explained about python programming language and developed code for the problem Statement.	48-53	PO3, PO4, PO5
Data Mining (C411)	C411.2	Students designed the modules for the solution of the problem.	33	PO2, PO3, PO4
Future Scope		Students explained about how they would like to further their project and develop it as their future scope.	61	PO12, PS02
Bibliography		Listed the references from which the literature was collected.	62	PO8, PO12
ENG		Prepared the thesis and intermediate progress reports and explained to the review panel. Also, continuously interact with guide and explain the progress.		PO9, PO10

SIGNATURE OF STUDENTS **SIGNATURE OF INTERNAL GUIDE**