**A Report**

**on**

**IMAGE STEGANOGRAPHY**

**Submitted for partial fulfilment of the requirements for the**

**Mini project Laboratory**

**of**

**BACHELOR OF ENGINEERING** **IN** **COMPUTER SCIENCE AND ENGINEERING**

**by**

**MOHD ABRAR AHMED 1604-19-733-050**

**AFSHAR SAMEEM 1604-19-733-053**

**ASHHAR ANIS 1604-19-733-056**

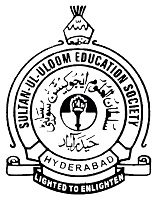
**Under the guidance of**

**Mr. Md. Zainuddin Naveed**

**Assistant Professor**

**Department of Computer Science and Engineering**

**MJCET, Hyderabad.**



**Department of Computer Science and Engineering**

**Muffakham Jah College of Engineering and Technology**

**Hyderabad-500034**

**CERTIFICATE**

This is to certify that the project work entitled “**IMAGE STEGANOGRAPHY”** is a bonafide work carried out by **Mr. MOHD ABRAR AHMED (1604-19-733-050), Mr. AFSHAR SAMEEM (1604-19-733-053), Mr. ASHHAR ANIS (1604-19-733-056)** in partial fulfilment of the requirements for the Mini project laboratory of **BACHELOR OF ENGINEERING IN COMPUTER SCIENCE AND ENGINEERING** by the **OSMANIA UNIVERSITY**, Hyderabad, under our guidance and supervision.

The results embodied in this report have not been submitted to any other university or institute for the award of any degree or diploma.

Course coordinator

ZAINUDDIN NAVEED

Department of CSE

MJCET, Hyderabad.

**DECLARATION**

This is to certify that the work reported in the mini project entitled “IMAGE STEGANOGRAPHY**”** is a record of work done by us in the Department of Computer Science and Engineering, Muffakham Jah College of Engineering and Technology, Osmania University. The reports are based on the project work done entirely by us and not copied from any other source.

**Mr. MOHD ABRAR AHMED**

**Mr. AFSHAR SAMEEM**

**Mr. ASHHAR ANIS**

**ACKNOWLEDGEMENT**

I would like to express my sincere gratitude and indebtedness to my project course coordinator **Mr. Md. Zainuddin Naveed** for his/her valuable suggestions and interest throughout the course of this project

I am also thankful to Head of the department **Dr.A.A.Moiz Qyser** for providing excellent infrastructure and a nice atmosphere for completing this project successfully

I convey my heartfelt thanks to the lab staff for allowing me to use the required equipment whenever needed

Finally, I would like to take this opportunity to thank my family for their support through the work. I sincerely acknowledge and thank all those who gave directly or indirectly their support in completion of this work

**MOHD ABRAR AHMED**

**AFSHAR SAMEEM**

**ASHHAR ANIS**

ABSTRACT

**OBJECTIVE**

* Hiding a secret message within a cover-media in such a way that others cannot discern the presence of the hidden message.
* Utilizing LSB (Least Significant Bit) method to hide data within an image.
* Formulating encoding/decoding functions to be shared between the sender/recipient.
* Implementing the whole process into a usable app.
* Constructing a simple and fluent UI.

**SOFTWARE REQUIREMENTS**

The software used in this project:

* Python
* Windows 7+
* Tkinter , stegano , PIL libraries/modules

**CONCLUSION**

The process of image steganography is simplified and made readily accessible for its users. The idea of hiding encrypted data within plain sight is implemented. Out of all the methods of Steganography, LSB is used and the extent of bit alteration is altered based upon amount of data to be hidden.

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE NUMBER** | **DESCRIPTION** | **PAGE NUMBER** |
| 2.1 | SIMILAR TOOLS COMPARISION | 7 |
| 3.1 | STAKEHOLDER | 8 |
| 3.2 | USECASE – DESCRIPTION ENCODING | 9 |
| 3.3 | USECASE – DESCRIPTION DECODING | 9 |
| 6.1 | TEST CASES | 22 - 23 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NUMBER** | **DESCRIPTION** | **PAGE NUMBER** |
| 1.1 | BASIC STEGANOGRAPHIC MODEL | 3 |
| 1.2 | RGB REPRESENTATION OF AN IMAGE | 3 |
| 1.3 | LSB MANIPULATION | 4 |
| 2.1 | STEGODOS APP WINDOW | 6 |
| 4.1 | DATA FLOW DIAGRAM | 11 |
| 4.2 | CLASS DIAGRAM | 12 |
| 4.3 | SEQUENCE DIAGRAM | 13 |
| 4.4 | USECASE DIAGRAM | 14 |

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| DESCRIPTION | PAGE NO |
| Certificate  Declaration  Acknowledgement  Abstract  List of Tables  List of Figures | i  ii  iii  iv  v  vi |
| CHAPTER 1 – INTRODUCTION   * 1. Objectives   2. Steganography      1. Steganography Types      2. Basic Steganographic model      3. Least Significant Bit Steganography | 1 |
| CHAPTER 2 – LITERATURE SURVEY  2.1 Similar system comparison  2.1.1 White Noise Storm  2.1.2 StegoDos  2.1.3 StegCure  2.2 Similar Tools comparision  2.3 Recommendations | 5 |
| CHAPTER 3 – SYSTEM ANALYSIS  3.1Introduction  3.2 Stakeholders  3.3 Non – Functional Requirements | 8 |
| CHAPTER 4 – SYSTEM DESIGN  4.1 Hardware/Software Requirements  4.2 UML Diagrams  4.2.1 Data Flow Diagrams  4.2.2 Class Diagram  4.2.3 Sequence Diagram  4.2.4 Use Case Diagram | 11 |
| CHAPTER 5 – IMPLEMENTATION  5.1 Login Screen  5.1.2 Invalid Login  5.2 Selecting the Functionality  5.3 Encode  5.3.1 Entering the Message  5.3.2 Selecting the File  5.3.3 Clicking the Encode Button  5.3.4 Checking Status  5.4 Decode  5.4.1 Selecting the Encoded File  5.4.2 Decoding the Image | 15 |
| CHAPTER 6 – RESULTS AND TESTING  6.1 Black Box Testing and White Box Testing  6.2 Test Cases | 22 |
| CHAPTER 7 – CONCLUSION & FUTURE ENHANCEMENTS  7.1 Conclusions  7.2 Future Enhancements | 24 |
| REFERENCES | 25 |

**CHAPTER 1**

**INTRODUCTION**

The growing use of Internet needs to take attention while we send and receive personal information in a secured manner. For this, there are many approaches that can transfer the data into different forms so that their resultant data can be understood if it can be returned back into its original form. This technique is known as encryption. However, a major disadvantage of this method is that the existence of data is not hidden. If someone gives enough time then the unreadable encrypted data may be converted into its original form.

A solution to this problem has already been achieved by using “steganography” technique to hide data in a cover media so that other cannot notice it. The characteristics of the cover media depends on the amount of data that can be hidden, the perceptibility of the message and its robustness.

* 1. **OBJECTIVES**

The project aims and objectives that will be achieved after completion of this are discussed in this subchapter. The aims and objectives are as follows:

* In this project we primarily concentrated on the data security issues when sending the data over the network using steganographic techniques.
* The main objectives of our project are to product security tool based on steganography techniques to hider message carried by stego-media which should not be sensible to human beings and avoid drawing suspicion to the existence of hidden message

**1.2 STEGANOGRAPHY**

Steganography is the process of hiding a secret message within a larger one in such a way that someone can not know the presence or contents of the hidden message. The purpose of Steganography is to maintain secret communication between two parties**.**

Hiding a message with Steganography methods reduces the chance of a message being detected. If the message is also encrypted then it provides another layer of protection.

Therefore, some Steganographic methods combine traditional Cryptography with Steganography; the sender encrypts the secret message prior to the overall communication process, as it is more difficult for an attacker to detect embedded cipher text in a cover. It has been used through the ages by ordinary people, spies, rulers, government, and armies.

Information hiding is a technique of hiding secret using redundant cover data such as images, audios, movies, documents, etc. This technique has recently become important in a number of application areas. For example, digital video, audio, and images are increasingly embedded with imperceptible marks, which may contain hidden signatures or watermarks that help to prevent unauthorized copy. It is a performance that inserts secret messages into a cover file, so that the existence of the messages is not apparent.

**1.2.1 STEGANOGRAPHY TYPES**

As it is known there is much communication between people and organizations through the use of the phone, the fax, computer communications, radio, and of course all of these communications should be secure. There are many types of Steganography some of which are

* Image Steganography
* Video Steganography
* Text Steganography
* Audio Steganography
* Network Steganography
* Email Steganography

**1.2.2 BASIC STEGANOGRAPHY MODEL**

As seen in the above image, both the original image file(X) and secret message (M) that needs to be hidden are fed into a steganographic encoder as input. Steganographic Encoder function, f(X,M,K) embeds the secret message into a cover image file by using techniques like least significant bit encoding. The resulting stego image looks very similar to your cover image file, with no visible changes. This completes encoding. To retrieve the secret message, stego object is fed into Steganographic Decoder.

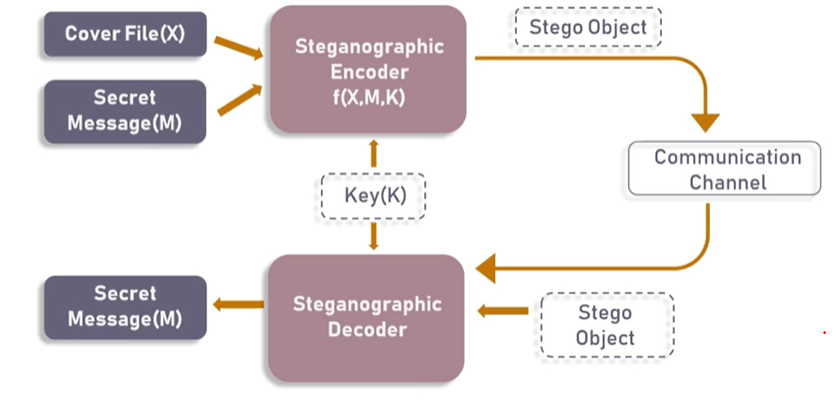
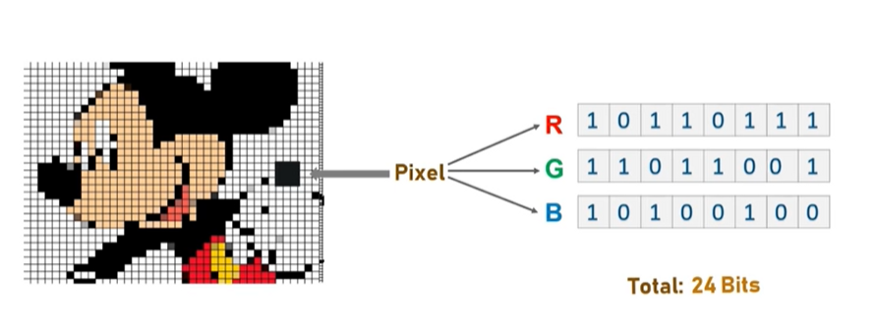


Fig 1.1 – Basic Steganographic Model

**1.2.3 LEAST SIGNIFICANT BIT STEGANOGRAPHY**

We can describe a **digital image** as a finite set of digital values, called pixels. Pixels are the smallest individual element of an image, holding values that represent the brightness of a given color at any specific point. So we can think of an image as a matrix (or a two-dimensional array) of pixels which contains a fixed number of rows and columns.

Least Significant Bit (LSB) is a technique in which the last bit of each pixel is modified and replaced with the secret message’s data bit.



1.2 – RGB Representation of an Image

From the next image it is clear that, if we change MSB it will have a larger impact on the final value but if we change the LSB the impact on the final value is minimal, thus we use least significant bit steganography.

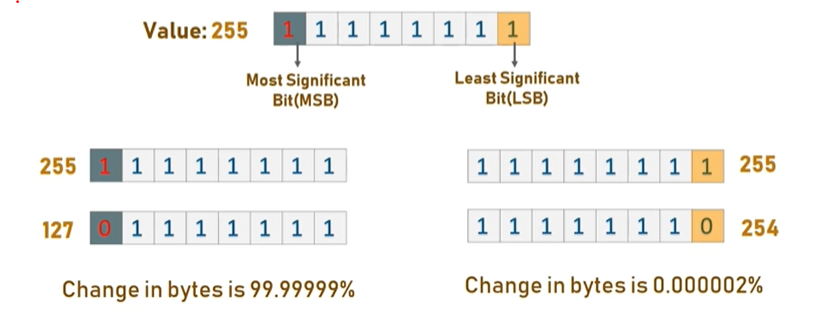


Fig 1.3 – LSB manipulation

**How LSB technique works?**

Each pixel contains three values which are Red, Green, Blue, these values range from **0 to 255**, in other words, they are 8-bit values. [4] Let’s take an example of how this technique works, suppose you want to hide the message “**hi**” into a **4x4** image which has the following pixel values:

**[(225, 12, 99), (155, 2, 50), (99, 51, 15), (15, 55, 22),(155, 61, 87), (63, 30, 17), (1, 55, 19), (99, 81, 66),(219, 77, 91), (69, 39, 50), (18, 200, 33), (25, 54, 190)]**

Using [the ASCII Table](http://www.asciitable.com/), we can convert the secret message into decimal values and then into binary: **0110100 0110101.**Now, we iterate over the pixel values one by one, after converting them to binary, we replace each least significant bit with that message bits sequentially (e.g 225 is 11100001, we replace the last bit, the bit in the right (1) with the first data bit (0) and so on).This will only modify the pixel values by +1 or -1 which is not noticeable at all. The resulting pixel values after performing LSBS is as shown below:

**[(224, 13, 99),(154, 3, 50),(98, 50, 15),(15, 54, 23),(154, 61, 87),(63, 30, 17),(1, 55, 19),(99, 81, 66),(219, 77, 91),(69, 39, 50),(18, 200, 33),(25, 54, 190)]**

**CHAPTER 2**

**LITERATURE SURVEY**

**INTRODUCTION**

In this chapter, we will provide an overview of steganography using LSB to hide the files inside images using python, it is used in a variety of domains, such as image and signal processing, Python offers many “libraries / modules”, and a simple interface to high-performance libraries, one of the most advantages is a large user community with lots of free code and knowledge sharing and the ability to process both still images and video. It is popular because of its ease and simplicity. Also, we will mention some programs that have the same approach it is using an encryption. Then, we will give the recommendation that help to develop our program.

In this project, use a method of encrypting any data file in an image file. This process of hiding the data helps to sharing the information with others over the internet network without any potential risk. The proposed system will help to hide the content with in the image and encryption of data file with in the image will help to make the document much securer.

In this research, developed the proposed system by using steganographic algorithm which is LSB and a technique for hiding capacity and efficiency of hiding the message with in an image.

**2.1 SIMILAR SYSTEMS COMPARISON**

**2.1.1 White Noise Storm**

White Noise Storm is a DOS based tool that could easily embed secret messages in cover images without any degradation. However, the integrity of the cover image could be severely affected by noise. The tool uses LSB steganography technique to embed secret messages in PCX files. The main disadvantage of this tool is the loss of many bits that can be used to hold information. Additionally, it uses large cover images to store information that it could be stored in a smaller cover images using other tools.

**2.1.2 StegoDos**

StegoDos is public domain software that only works on 320x200 pixel images with 256 colors. The tool uses LSB steganography technique to hide secret messages. The main disadvantage of the tool is the size restriction that limits the user’s cover image to 320x200 pixels in-order to have a stego-image that is similar to the original one. Another disadvantage is the dependence on the end-of- file character to end the message that does not have any significance work since the message after retrieval appears to contain garbage

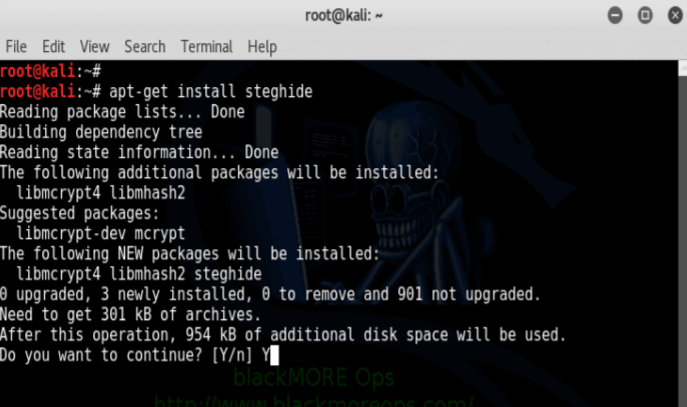


Fig 2.1 – StegoDos app window

**2.1.3 StegCure**

StegCure uses three different LSB steganography techniques. In compared with the other tools, StegCure offers a better security and has a user-friendly functionality with interactive graphical user interface (GUI) and integrated navigation capabilities. Also, it can prevent any attacks by restricting the user to one attempt to retrieve the secret message.

**2.2 SIMILAR TOOLS COMPARISON**

Table in the next page shows a comparison between our proposed system tool and other image steganography tools.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **White Noise Storm** | **StegoDos** | **StegCure** | **Proposed System** |
| **Capacity of Secret Message** | Limited hiding capacity | Limited hiding capacity | Limited hiding capacity | Optimum hiding capacity |
| **Image Size** | - | 320 x 600 | - | Variable size |
| **Image Format** | PCX | Lossless (Ex. GIF and BMP) | GIF | JPEG / JPG |
| **Efficiency** | Low | Low | Low | Medium |

Table 2.1 – Similar Tools Comparision

2.3 RECOMMENDATIONS

After studying similar tools to our proposed system LSB substitution is used to embed the message into an image. It works by adjusting the LSB of the carrier image’s pixels whereas, the last bit of each byte in the image is changed to a bit of the secret message that is known standard LSB (1-LSB). Also, use 2-LSB method that differs from the standard LSB method by allowing more data to be hidden into the cover image. The idea of this method is almost similar to the standard LSB, except that it replaces the 2-LSB of each byte of the cover image instead of one bit. The LSB insertion differs depending on the number of bits in an image. In 8-bit images, the last bit of each byte in the image is changed to a bit of the secret message. However, it has a major limitation, which is embedding only small size data into images. While in 24- bit images, the last bit of each RGB component is changed which allows more data to be hidden.

**CHAPTER 3**

**SYSTEM ANALYSIS**

**3.1 INTRODUCTION**

After analyzing the requirements of the task to be performed, the next step is to analyze the problem and understand its context. The first activity in the phase is studying the existing system and other is to understand the requirements and domain of the new system. Both the activities are equally important, but the first activity serves as a basis of giving the functional specifications and then successful design of the proposed system. Understanding the properties and requirements of a new system is more difficult and requires creative thinking and understanding of existing running system is also difficult, improper understanding of present system can lead diversion from solution.

**3.2 STAKEHOLDERS**

Stakeholders must be able to make decisions, but need not be human: "An stakeholder might be a person, a company or organization, a computer program, or a computer system — hardware, software, or both." Actors are always stakeholders, but not all stakeholders are actors.

|  |  |
| --- | --- |
| **STAKEHOLDER** | **DESCRIPTION** |
| User | That have major controlling of the whole systems process and able to access or use all modules of the application. |
| Program | The Application itself which can perform the user operation in main process encrypt ( hide image in another image), decrypt (unhide image from Stego-image). |

Table 3.1 Stakeholders

* **Choose an image -** Select the picture in which the data is to be hidden
* **Encoding -** Encode the image with the data which is to be hidden
* **Check Status -** Check whether the image is encoded or not
* **Decoding –** Extract the hidden data from the image

|  |  |
| --- | --- |
| Actors | user |
| Description | The user wants to hide data in an image file |
| data | Message , Cover Image |
| response | 1. Select the file 2. Enter the data which is to be hidden 3. Click on the Encode button |
| comments | In case the user doesn’t enter the message the app alerts the user to enter the message |
| stimuli | Start hiding process by clicking the encode button |

Table 3.2 Use case description - Encoding process

|  |  |
| --- | --- |
| Actors | user |
| Description | The user wants to extract data from the image file |
| data | Copy Image |
| response | 1. Select the copied file 2. Click on the Decode button 3. Display the message |
| comments | In case the user doesn’t choose an appropriate file the application alerts the user to choose an appropriate file |
| stimuli | Start decoding process by clicking the decode button |

Table 3.3 Use case description – Decoding process

**3.4 NON-FUNCTIONAL REQUIREMENTS**

• For user interfaces we take in consideration that they should has a standard look and

being user friendly at the same time to make sure that users' attention will not be

distracted and interface to provide more flexibility and scalability.

• The program will be in the English language

• The program must be fast in processing

• The program must to hide the message within the image and then extract the message from the image properly.

• All function must be works well then system will be a high quality

**CHAPTER – 4**

**SYSTEM DESIGN**

**4.1 HARDWARE / SOFTWARE REQUIREMENTS**

The execution phase was developed based upon two phases. The different phases are encryption and decryption phases. We require few hardware and software interfaces for implementing these phases.

* Any code editor (VS code)
* Windows 7+
* Installation of tkinter library
* Installation of Pillow library
* Installation of PIL library

**4.2 UML DIAGRAMS**

UML diagrams are used to describe the relationships between classes. Structural view can be described using class diagram.

**4.2.1 Data Flow Diagram**

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design). A DFD shows what kinds of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel.



Fig 4.1 – Data Flow Diagram

**4.2.2 CLASS DIAGRAM**

A class diagram is a picture for describing generic descriptions of possible systems. Class diagrams and collaboration diagrams are alternate representations of object models. Class diagrams contain classes and object diagrams contain objects, but it is possible to mix classes and objects when dealing with various kinds of metadata, so the separation is not rigid.

Class diagrams are more prevalent than object diagrams. Normally you will build class diagrams plus occasional object diagrams illustrating complicated data structures or message-passing structures.

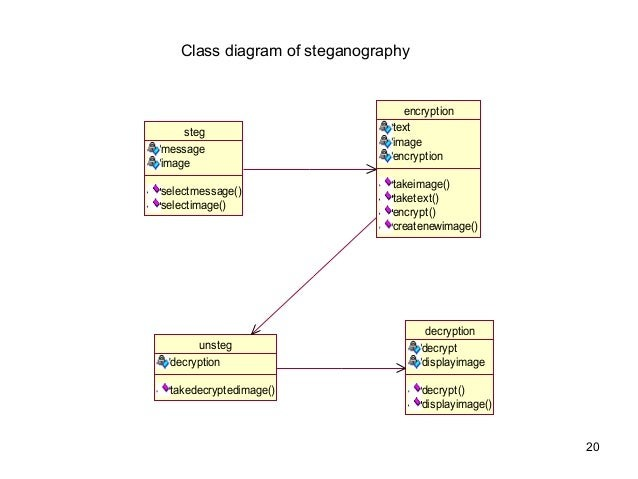


Fig 4.2 – Class Diagram

**4.2.3 SEQUENCE DIAGRAM**

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

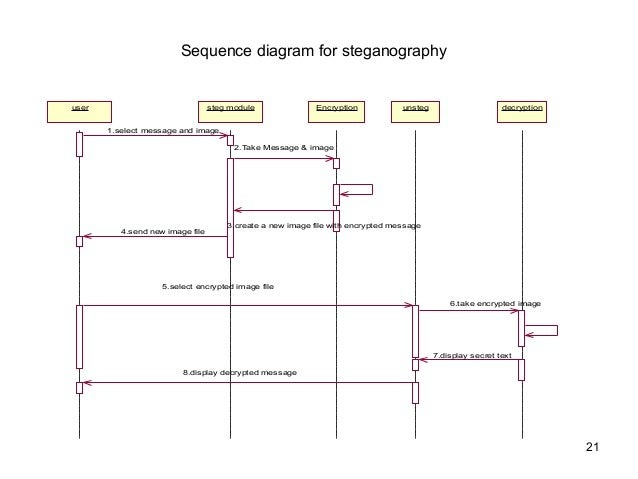


Fig 4.3 – Sequence Diagram

**4.2.4 USECASE DIAGRAM**

A use case is a set of scenarios that describing an interaction between a user and a system. A use case diagram displays the relationship among actors and use cases

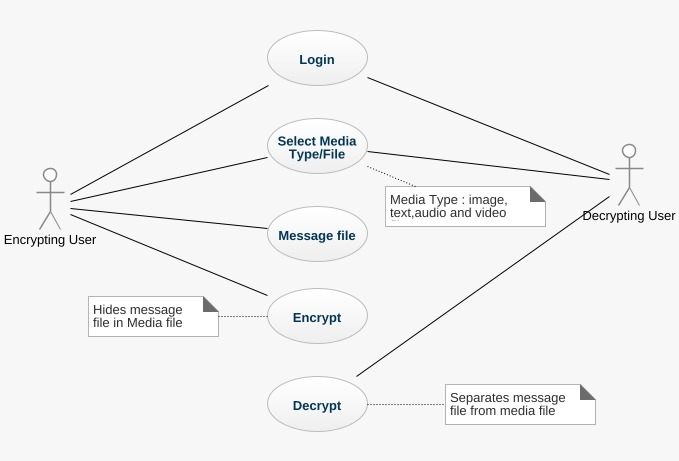
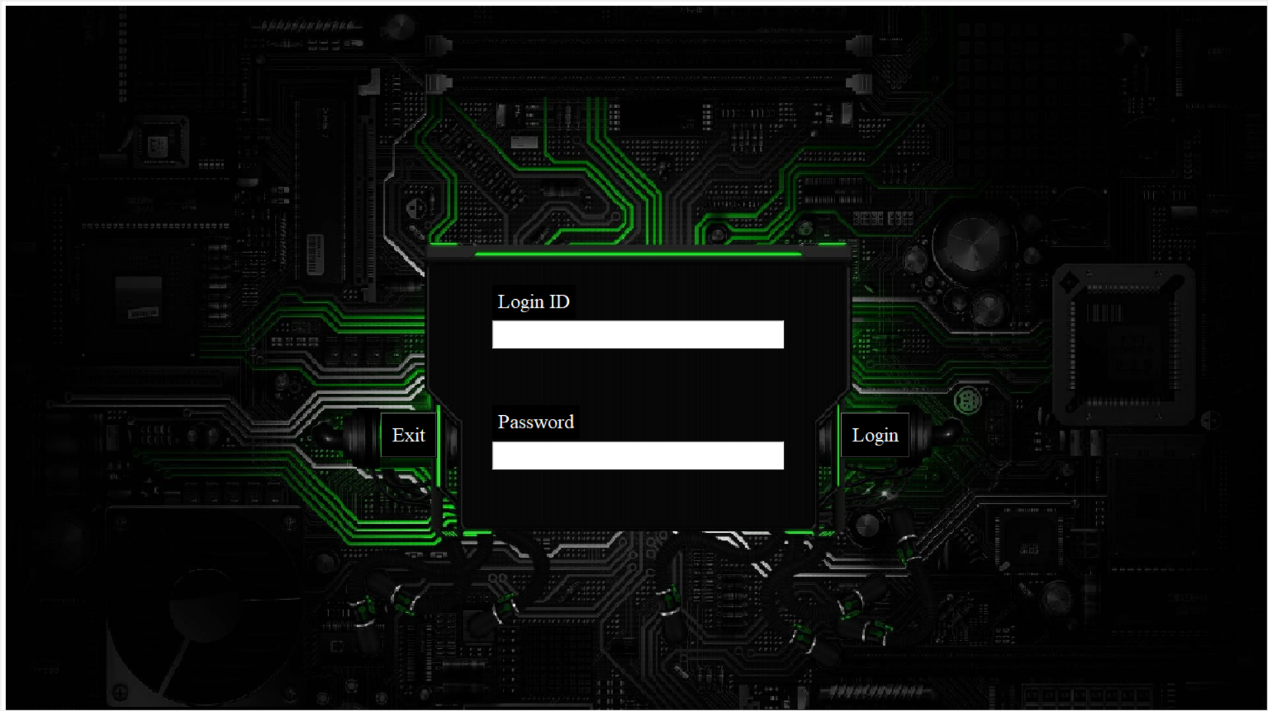
****

Fig 4.4 - Use case Diagram

**CHAPTER 5**

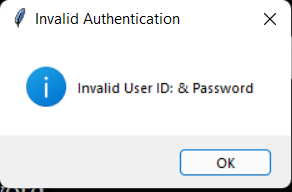
**IMPLEMENTATION**

**5.1 LOGIN SCREEN**

****

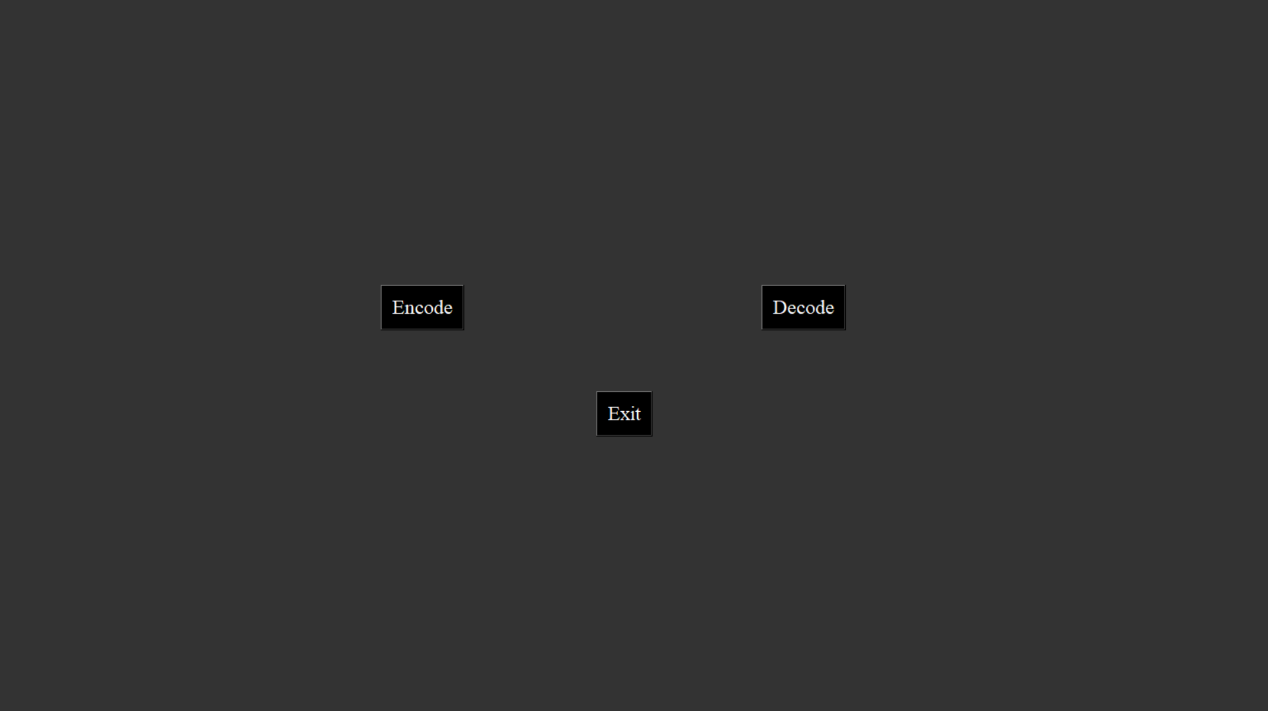
The user has to enter his login credentials to use the application

**5.1.2 INVALID LOGIN**

****

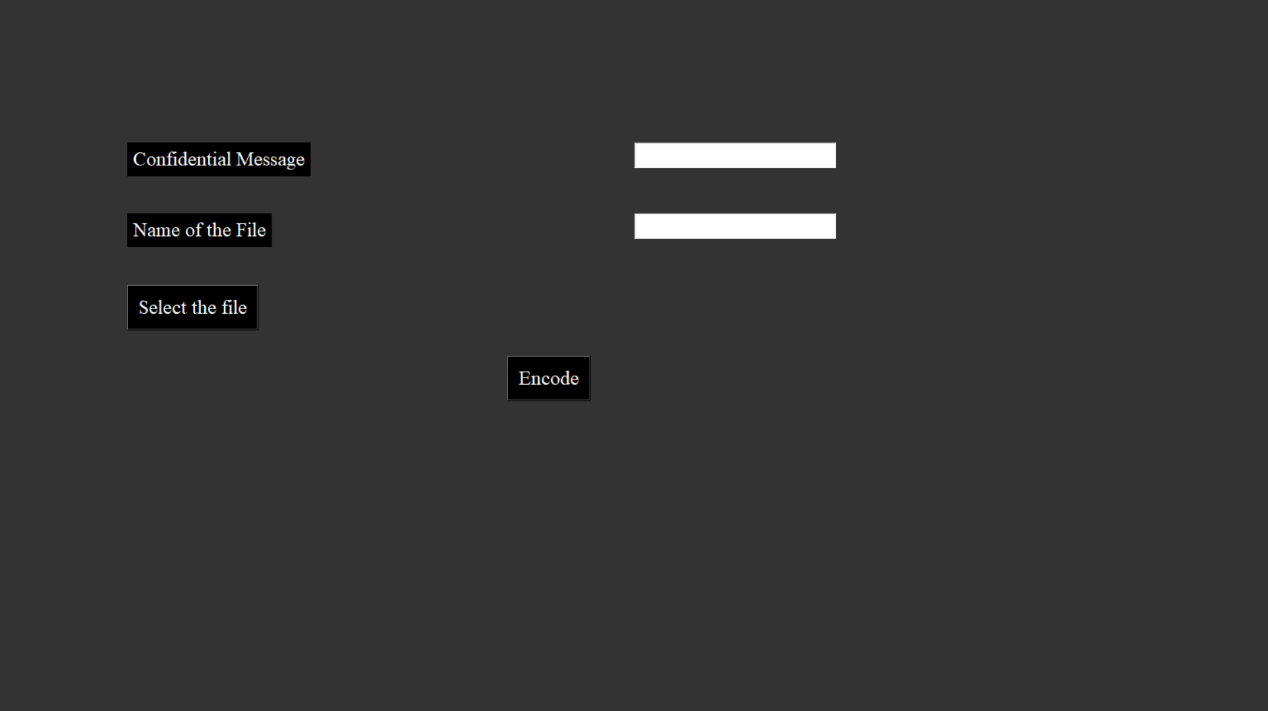
This message box is prompted in case of an invalid login

**5.2 SELECTING THE FUNCTIONALITY**

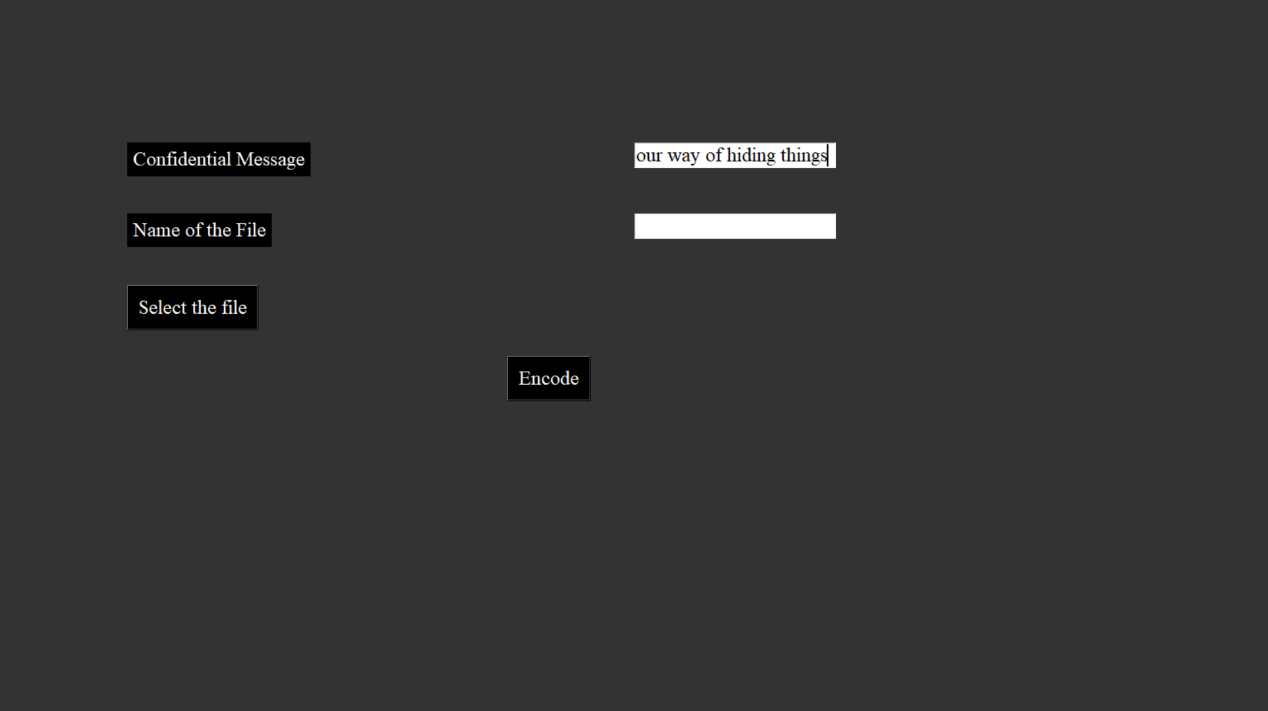
****

Here the user selects whether to encode or decode.

**5.3 ENCODE**

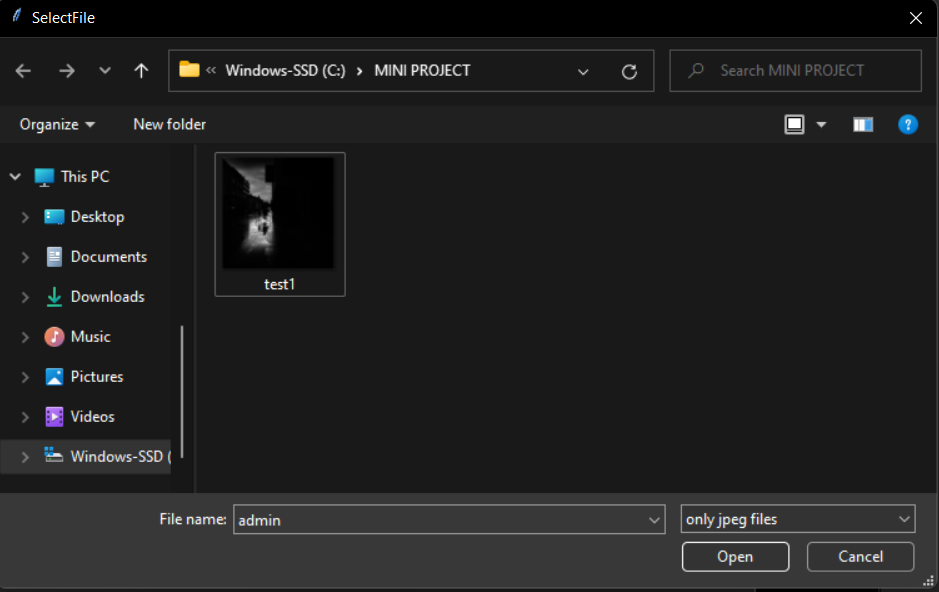
****

**5.3.1. ENTERING THE MESSAGE**

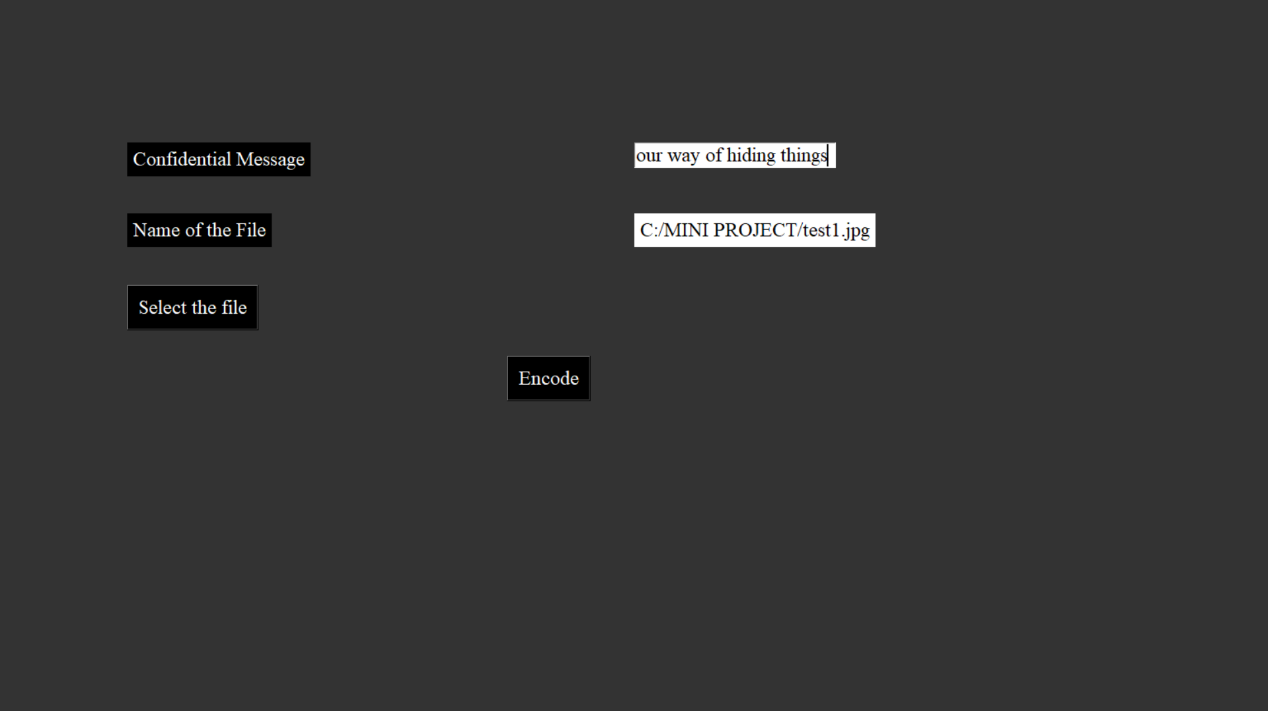
****

The user enters the data that the user wants to hide in the image

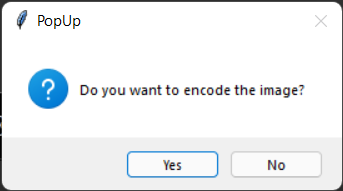
**5.3.2. SELECTING THE FILE**

****

By clicking on the select file button, we can browse the file from the system.

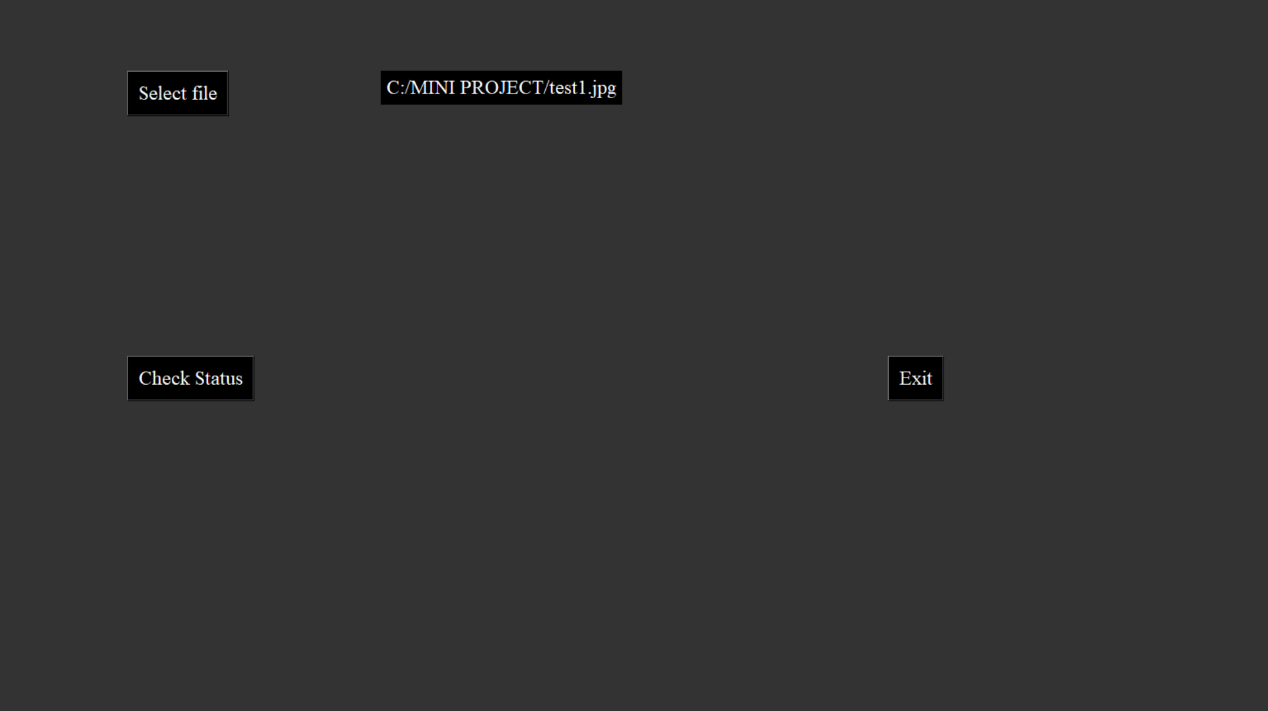


**5.3.3. CLICKING THE ENCODE BUTTON**

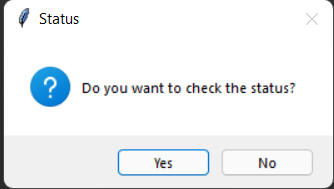
****

If the user selects yes the application navigates to a new screen which checks the status of the file.

**5.3.4 CHECKING STATUS**

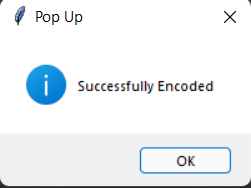


Here the user selects the file again from the system and clicks the check status button it gives a popup to confirm.

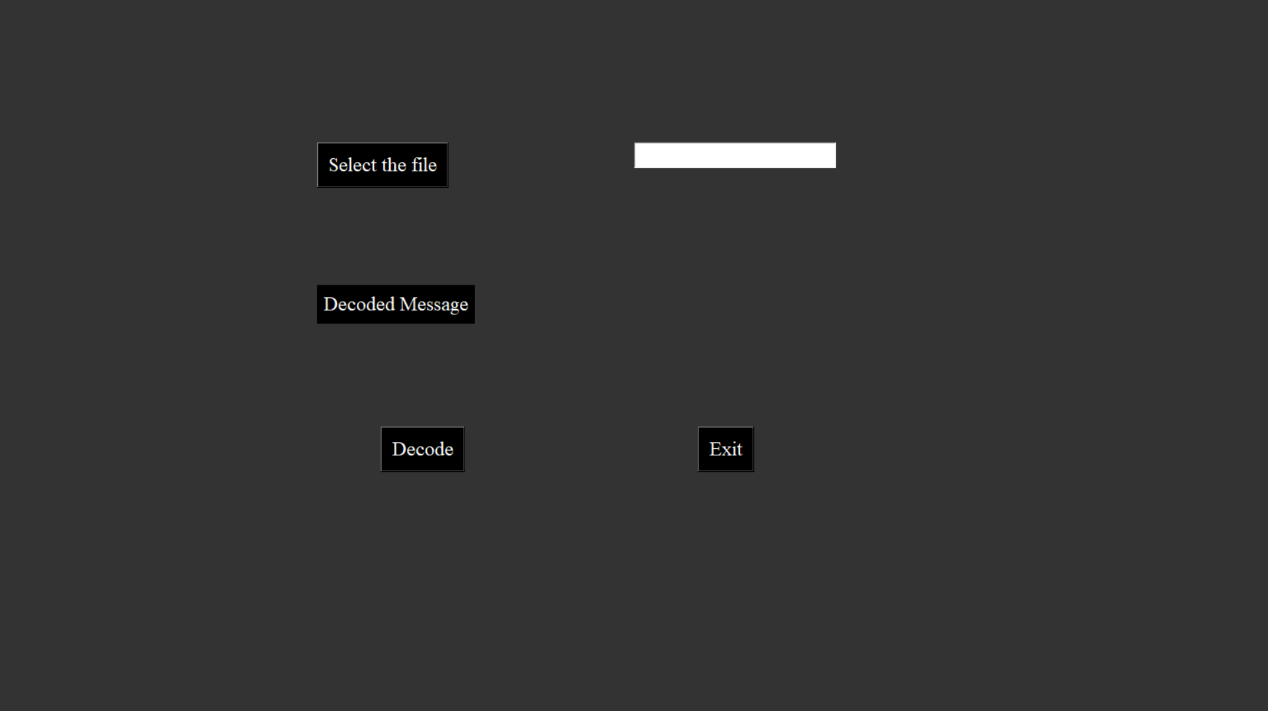


The user gets a popup on successful encryption of the image

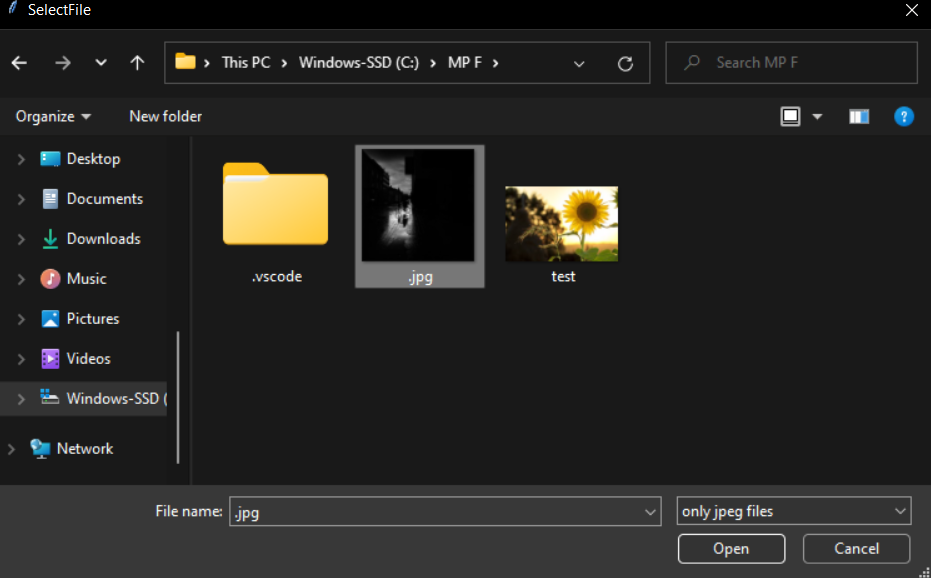
After the successful encoding the encoded image is generated.



**5.4 DECODE**

****

**5.4.1 SELECTING THE ENCODED FILE**

****

****

**5.4.2 DECODING THE IMAGE**

****

After clicking the decode button the hidden data in the image is decoded and displayed.

**CHAPTER 6**

**RESULTS AND DISCUSSIONS**

**6.1 BLACK BOX AND WHITE BOX TESTING**

Black box testing is the testing approach which tells us about the possible combinations for the end-user action. Black box testing doesn't need the knowledge about the interior connections or programming code. In the black box testing, the user tests the application by giving different sources and checks whether the output for the specified input is appropriate or not.

White box testing is also known as "glass box" or "clear box" or "open box" testing. It is opposite to the black box testing. In the white box testing, we can create test cases by checking the code and executing in certain intervals and know the potential errors. The analysis of the code can be done by giving suitable inputs for the specified applications and using the source code for the application blocks.

**6.2 TEST CASES**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STEP | TEST CONDITION | ACTION | EXPECTED RESULT | ACTUAL RESULT |
| 1.LOGIN | Check valid id & password | Click login | Navigate to home screen | Done |
| 2.ENCODE |  |  |  |  |
| 2.1. | Enter the message | Input | Should display the typed message | Done |
| 2.2 | Select the image | Click | Display the path of image | Done |
| 2.3 | Click Encode button | Click | Check Status | Done |
| 3. Decode |  |  |  |  |
| 3.1 | Select the image | Click | Display the path of image | Done |
| 3.2 | Click Decode button | Click | Display the encoded message | Done |

Table 6.1 – Test Cases

**CHAPTER 7**

**CONCLUSION & FUTURE ENHANCEMENTS**

**7.1 CONCLUSIONS**

Although only some of the main image steganographic techniques were discussed in this document, one can see that there exists a large selection of approaches to hiding information in images. All the major image file formats have different methods of hiding messages, with different strong and weak points respectively. Where one technique lacks in payload capacity, the other lacks in robustness. For example, the patchwork approach has a very high level of robustness against most type of attacks, but can hide only a very small amount of information.

Least significant bit (LSB) in both BMP and GIF makes up for this, but both approaches result in suspicious files that increase the probability of detection when in the presence of a warden.

The proposed approach in this project uses a new steganographic approach called image steganography. The application creates a stego image in which the personal data is embedded inside the cover file image.

Used the Least Significant Bit algorithm in this project for developing the application which is faster and reliable and compression ratio is moderate compared to other algorithms.

**7.2 FUTURE ENHANCEMENTS**

The major limitation of the application is designed for images cover files. It accepts only images as a carrier file.

The future work on this project is to improve the compression ratio of the image to the text. This project can be extended to a level such that it can be used for the different types of multimedia files

**REFERENCES**

[1] Rosziati Ibrahim and Teoh Suk Kuan, Steganography Imaging System (SIS): Hiding

Secret Message inside an Image

<http://www.iaeng.org/publication/WCECS2010/WCECS2010_pp144-148.pdf>

[2] Zaidoon Kh. AL-Ani, A.A.Zaidan, B.B.Zaidan and Hamdan.O.Alanazi, Overview: Main Fundamentals for

Steganography

<http://arxiv.org/ftp/arxiv/papers/1003/1003.4086.pdf>

[3] Data Flow Diagram Symbols

<http://www.idi.ntnu.no/~sif8035/pdf/flesn/notation.pdf>

[4] Donald S. Le Vie, Jr., Understanding Data Flow Diagrams

<http://ratandon.mysite.syr.edu/cis453/notes/DFD_over_Flowcharts.pdf>

[5] B. Beizer, Black Box Testing. New York: John Wiley & Sons, Inc., 1995.

[6] A. Bertolino, "Chapter 5: Software Testing," in IEEE SWEBOK Trial Version

1.00, May 2001.

[7] Testing Overview and Black-Box Testing Techniques

<http://agile.csc.ncsu.edu/SEMaterials/BlackBox.pdf>

[8] Jovanović, Irena , Software Testing Methods and Techniques,

<http://www.internetjournals.net/journals/tir/2009/January/Paper%2006.pdf>