
# J.B.S.P SANSTHA'S
# CHANGU KANA THAKUR
# ARTS, COMMERCE, SCIENCE COLLEGE
# NEW PANVEL (W)

**PROJECT REPORT**
**ON**

# STEGANOTECH – A STEGANOGRAPHY TOOL

**DEVELOPED BY**
## Mr. PAWAN A. GOSAVI

**UNDER THE GUIDANCE OF**
## PROF. MISS. AISHWARYA R. KADAM

**SUBMITTED TO**
## UNIVERSITY OF MUMBAI
## 2018 – 2019
**For V$^{th}$ SEMESTER**


**J.B.S.P SANSTHA'S**
**CHANGU KANA THAKUR**
**ARTS, COMMERCE, SCIENCE COLLEGE**
**NEW PANVEL (W)**

**PROJECT REPORT**
**ON**

# STEGANOTECH

A Steganography Tool

**SUBMITTED TO**
## UNIVERSITY OF MUMBAI
## 2018 – 2019
**For V$^{th}$ SEMESTER**

**BY**
## Mr. PAWAN A. GOSAVI

**UNDER THE GUIDANCE OF**
## PROF. MISS. AISHWARYA R. KADAM

# DEPARTMENT OF COMPUTER SCIENCE

## CERTIFICATE

This is to certify that the project entitled

**“STEGANOTECH – A STEGANOGRAPHY TOOL”**

Is successfully completed by **Mr. PAWAN A. GOSAVI**, Roll No: CS13. Examination Number _____ under the guidance of **PROF. MISS. AISHWARYA R. KADAM**, during the academic period of 27th August 2018 to 10th October 2018 as per the Syllabus, and the fulfillment for the completion of the B.Sc. degree in the Computer Science of University of Mumbai. It is also to certify that this is original work of the candidate done during academic year 2018 - 2019.

**Place:**

**Date:**

**Internal Examiner** **Head of Department** **Principal**

**External Examiner**

# Acknowledgment

It is with the deep sense of gratitude that we acknowledge the help received from many quarters in completion of this project. There are many people without whom this unique learning experience would be a non-starter.

I would like to express gratitude towards respected **Prof. Mrs. Pratibha Jadhav**, (Head of Computer Science department, CKT college, New Panvel) for encouraging us to do hard work and devote ourselves sincerely to the project.

The completion of the project work is a milestone in student life and its execution is inevitable in the hands of guide. A work of this type could not have been possible without consulting and advice of **Prof. Ms. Aishwarya R. Kadam**, and our subject **Prof. Mrs. Pratibha Jadhav**. I would like to thank them for their valuable guidance and appreciation for giving form and substance to this report. It is due to their enduring efforts; patients and enthusiasm, which has given a sense of direction and purpose fullness to this project and ultimately made it a success.

I would also like to express our deep regards and gratitude to the Principal Dr. S. T. Gadade.

Also, thanks to Non-Teaching staff for providing help in our project completion. We owe more than we express to our parents and well-wishers without their encouragement this project would not have taken this shape.

# Index

# ABSTRACT

Steganography is the art of hiding the fact that communication is taking place, by hiding information in other information. Many different carrier file formats can be used, but digital images are the most popular because of their frequency on the internet. For hiding secret information in images, there exists a large variety of steganography techniques some are more complex than others and all of them have respective strong and weak points. Different applications may require absolute invisibility of the secret information, while others require a large secret message to be hidden. This project report intends to give an overview of image steganography, its uses and techniques. It also attempts to identify the requirements of a good steganography algorithm and briefly reflects on which Steganographic techniques are more suitable for which applications.

Steganography is the art and science of writing hidden messages in such a way that no one, apart from the sender and intended recipient, suspects the existence of the message, a form of security through obscurity. Steganography comes from the Greek and literally means, “Covered writing". It is one of various data hiding techniques, which aims at transmitting a message on a channel where some other kind of information is already being transmitted. This distinguishes steganography from covert channel techniques, which instead of trying to transmit data between two entities that were unconnected before.

The goal of steganography is to hide messages inside other “harmless” messages in a way that does not allow any “enemy” to even detect that there is a second secret message present. The only missing information for the “enemy” is the short easily exchangeable random number sequence, the secret key, without the secret key, the “enemy” should not have the slightest chance of even becoming suspicious that on an observed communication channel, hidden communication might take place. Steganography is closely related to the problem of “hidden channels” n secure operating system design, a term which refers to all communication paths that cannot easily be restricted by access control mechanisms. In an ideal world we would all be able to send openly encrypted mail or files to each other with no fear of reprisals. However there are often cases when this is possible, either because the working company does not allow encrypted email or the local government does not approve of encrypt communication (a reality in some parts of the world). This is where steganography can come into play.

# SYNOPSIS

**Project Name:** **STEGANOTECH – A STEGANOGRAPHY TOOL**

**Project Members:**

This project is an Individual Project. Project is done by **Mr. Pawan Gosavi.**

# INTRODUCTION

SteganoTech is software, which tries to alter the BMP file properties to allow the encrypted text document to be embedded into it. The text file is first compressed, encrypted and then embedded into the BMP file to allow maximum performance and robustness. This allows the users to easily and securely carry the compressed data. The major task of the BMP Steganography is to provide the user the flexibility of passing the information implementing the encryption standards as per the specification and algorithms proposed and store the information in a form that is undetectable. This Application has a reversal process, which is used to de-embed the data file from BMP file and decrypt the data to its original format upon the proper request by the user. While the Encryption and Decryption is done the application should satisfy the standards of authentication and authorization of the user.

The Entire application provides a user friendly Graphical User Interface, which is in a self-learning mode for the end user. The System will provide all the functional standards 2 of proper navigation within the environment, which makes it possible for the users to have a smooth flow while working with this environment. The Application is designed in such a way that, as soon as it creates Buffer for compressed data, the application asks the user for the Encryption Key details and starts its functionality upon the logistics that are provided with in this key. The key should be given in such a way that it should prevent unauthorized persons from identifying the encrypted information at any point of time. This application provides De-embedding, Decryption and Uncompressing for reversal process, which is carried at the other end, and the receiver will be able to Decrypt and Uncompressed the data only if the correct key is entered.

One of the reasons that intruders can be successful is the most of the information they acquire from a system is in a form that they can read and comprehend. Intruders may reveal the information to others, modify it to misrepresent an individual or organization, or use it to launch an attack. One solution to this problem is, through the use of steganography. Steganography is a technique of hiding information in digital media. In contrast to cryptography, it is not to keep others from knowing the hidden information but it is to keep others from thinking that the information even exists.

Steganography become more important as more people join the cyberspace revolution. Steganography is the art of concealing information in ways that prevents the detection of hidden messages. Steganography include an array of secret communication methods that hide the message from being seen or discovered.

Due to advances in ICT, most of information is kept electronically. Consequently, the security of information has become a fundamental issue. Besides cryptography, steganography can be employed to secure information. In cryptography, the message or encrypted message is embedded in a digital host before passing it through the network, thus the existence of the message is

unknown. Besides hiding data for confidentiality, this approach of information hiding can be extended to copyright protection for digital media: BMP, video and images.

The growing possibilities of modern communications need the special means of security especially on computer network. The network security is becoming more important as the number of data being exchanged on the internet increases. Therefore, the confidentiality and data integrity are requires to protect against unauthorized access and use. This has resulted in an explosive growth of the field of information hiding. Information hiding is an emerging research area, which encompasses applications such as copyright protection for digital media, watermarking, fingerprinting, and steganography. In watermarking applications, the message contains information such as owner identification and a digital time stamp, which usually applied for copyright protection. Fingerprint, the owner of the data set embeds a serial number that uniquely identifies the user of the data set. This adds to copyright information to makes it possible to trace any unauthorized use of the data set back to the user.

Steganography hide the secrete message within the host data set and presence imperceptible and is to be reliably communicated to a receiver. The host data set is purposely corrupted, but in a covert way, designed to be invisible to an information analysis.

## What is Steganography?

**Steganography** is the practice of hiding private or sensitive information within something that appears to be nothing out to the usual. Steganography is often confused with cryptology because the two are similar in the way that they both are used to protect important information. The difference between two is that steganography involves hiding information so it appears that no information is hidden at all. If a person or persons views the object that the information is hidden inside of he or she will have no idea that there is any hidden information, therefore the person will not attempt to decrypt the information.

What steganography essentially does is exploit human perception, human senses are not trained to look for files that have information inside of them, although this software is available that can do what is called Steganography. The most common use of steganography is to hide a file inside another file.

Information hiding has recently gained importance in various applications. One such area where information hiding is important is military. Steganography has become one of the popular approaches for information hiding. A recent breakthrough in this field is hiding information in music. The embedded data should be as immune as possible to modifications from intelligent attacks or anticipated manipulations. Thus the hidden messages are encrypted before hiding behind BMP files. Goal of this project is to hide encrypted information in music.

"In steganography "cover" is the medium which is used to hide the secret information. The information to be hidden can be a plain text message, a cipher text, another image, or anything that can be represented in binary." An image, BMP file or a video file usually act as cover to hold the information. If an image is used as cover then it can be altered in the noisy areas with a lot of color variations so that the alterations are less obvious. The message can also be scattered randomly throughout the image.

Common methods of concealing data in digital images include: Least significant bit insertion: This is a very simple method of hiding the message in a digital image. In this method, the LSB of each byte in the image is used to store the secret data. Changes in new image in which the message is hidden are too small to be recognized by the human eye. The disadvantage of this technique is that since it uses each pixel in an image, a lossless compression format like bmp or gif has to be used for the image.

Masking and filtering: "These methods hide information in a manner similar to paper watermarks. This can be done, for example, by modifying the luminance of parts of the image. It does change the visible properties of an image, but if done with care the distortion is barely discernable." Transformations: This is a complex way of hiding information in an image. This is considered as the most efficient way of hiding the information. Various algorithms and transformations are applied on the image to hide information in it. DCT (Direct cosine transformation) is one such method.

## History of Steganography

Throughout history Steganography has been used to secretly communicate information between people.

Some examples of use of Steganography is past times are:

1. During World War 2 invisible ink was used to write information on pieces of paper so that the paper appeared to the average person as just being blank pieces of paper. Liquids such as milk, vinegar and fruit juices were used, because when each one of these substances are heated they darken and become visible to the human eye.

2. In Ancient Greece they used to select messengers and shave their head, they would then write a message on their head. Once the message had been written the hair was allowed to grow back. After the hair grew back the messenger was sent to deliver the message, the recipient would shave off the messengers hair to see the secrete message.

3. Another method used in Greece was where someone would peel wax off a tablet that was covered in wax, write a message underneath the wax then re-apply the wax. The recipient of the message would simply remove the wax from the tablet to view the message.

## Why This Steganography?

This technique is chosen, because this system includes not only imperceptibility but also un-delectability by any steganolysis tool.

## Project Scope:

This project is developed for hiding information or text in any image file. The scope of the project is implementation of steganography tools for hiding information includes any type of information file and image files and the path where the user wants to save Image and extruded file.

## Methodology:

User needs to run the application. The user has two tab options for Both Steganography Types – encrypt and decrypt. If user select encrypt, application give the screen to select image file or Cover File, information file or Text, Creating Password and option to save the image file or Stego File. If user select decrypt, application gives the screen to select only image file and ask path where user want to save the secrete file also Ask for Password.

This project has two methods – Encrypt and Decrypt. In encryption the Secret information is hiding in with any type of image file. Decryption is getting the Secret information from image file.

## Problem Statement:

The former consists of linguistic or language forms of hidden writing. The later, such as invisible ink, try of hide messages physically. One disadvantage of linguistic steganography is that users must equip themselves to have a good knowledge of linguistry.

In recent years, everything is trending toward digitization. And with the development of the internet technology, digital media can be transmitted conveniently over the network. Therefore, messages can be secretly carried by digital media by using the steganography techniques, and then be transmitted through the internet rapidly. Steganography is the art of hiding the fact that communication is taking place, by hiding information in other information. Many different carrier file formats can be used, but digital images are the most popular because of their frequency on the internet.

For hiding secret information in images, there exists a large variety of steganography techniques some are more complex than others and all of them have respective strong and weak points. So we prepare this application, to make the information hiding simpler and user friendly.

## Cryptography:

"On the other hand cryptography is the process of conversion of data into scrambled code that can be deciphered and sent across a public or private network". The two main forms of encrypting data in cryptography are symmetrical and asymmetrical. Symmetric encryptions, or algorithms, use the same key for encryption as they do for decryption. Other names for this type of encryption are secret-key, shared key, and private-key. Symmetric cryptography is at times simple to decode.

Asymmetric cryptography uses different encryption keys for encryption and decryption. In we have one key for encryption and a different key for decryption. These keys are labeled or known as a public and a private key; in this instance the private key cannot be derived from the public key. The asymmetrical cryptography method has been proven to be secure as compared to symmetric cryptography. The most common form of asymmetrical encryption is in the application of sending messages where the sender encodes and the receiving party decodes the message by using a random key generated by the public key of the sender.

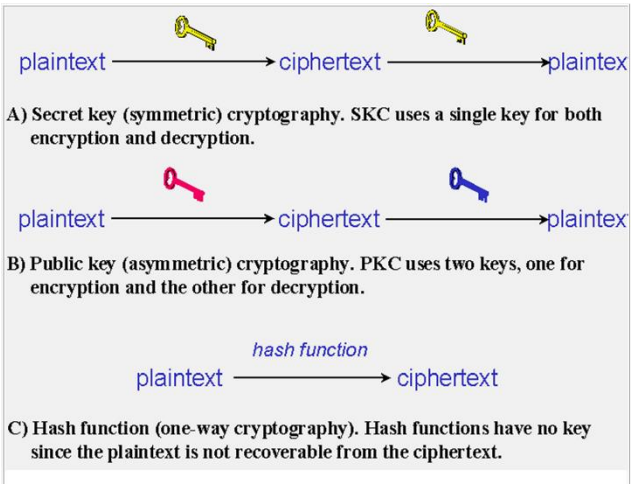
## Types of Algorithms

Cryptographic algorithms can be categorized in several ways. One of the ways is that they can be categorized based on the number of keys that are employed for encryption and decryption, and further defined by their application and use. The three types of algorithms that will be discussed, as shown in which is taken from are:

1. Secret Key Cryptography (SKC): Uses a single key for both encryption and decryption.

2. Public Key Cryptography (PKC): Uses one key for encryption and another for decryption.

3. Hash Functions: Uses a mathematical transformation to encrypt information.



## Steganography vs Cryptography:

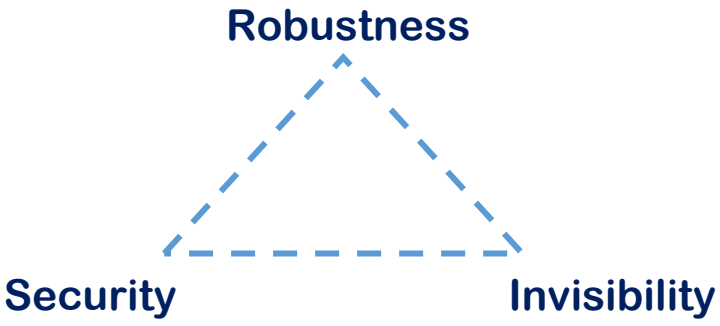
Basically, the purpose of cryptography and steganography is to provide secret communication. However, steganography is not the same as cryptography. Cryptography hides the contents of a secrete message from a malicious people, whereas steganography even conceal the existence of the message. In cryptography, the system is broken when the attacker can read the secret message. Breaking a steganography system need the attacker to detect that steganography has been used.

It is possible to combine the techniques by encrypting message using cryptography and then hiding the encrypted message using steganography. The resulting Stego-image can be transmitted without revealing that secret information is being exchanged.

## Steganography vs watermarking:

Steganography pay attention to the degree of Invisibility while watermarking pay most of its attribute to the robustness of the message and its ability to withstand attacks of removal, such as image operations(rotation, cropping, filtering), BMP operations(rerecording, filtering)in the case of images and BMP files being watermarked respectively.

It is a non-questionable fact that delectability of a vessel with an introduced data (steganographic message or a watermark) is a function of the changeability function of the algorithm over the vessel.




That is the way the algorithm changes the vessel and the severity of such an operation determines with no doubt the delectability of the message, since delectability is a function of file characteristics deviation from the norm, embedding operation attitude and change severity of such change decides vessel file delectability.

A typical triangle of conflict is message Invisibility, Robustness, and Security. Invisibility is a measure of the in notability of the contents of the message within the vessel.

Security is sin ominous to the cryptographic idea to message security, meaning inability of reconstruction of the message without the proper secret key material shared. Robustness refers to the endurance capability of the message to survive distortion or removal attacks intact. It is often used in the watermarking field since watermarking seeks the persistence of the watermark over attacks, steganographic messages on the other hand tend to be of high sensitivity to such attacks. The more invisible the message is the less secure it is (cryptography needs space) and the less robust it is (no error checking/recovery introduced).The more robust the message is embedded the more size it requires and the more visible it is.

## Steganography Techniques:

Over the past few years, numerous steganography techniques that embed hidden messages in multimedia objects have been proposed. There have been many techniques for hiding information or messages in images in such a manner that alteration made to the image is perceptually indiscernible. Commonly approaches are include LSB, Masking and filtering and Transform techniques.

Least significant bit (LSB) insertion is a simple approach to embedding information in image file. The simplest steganography techniques embed the bits of the message directly into least significant bit plane of the cover-image in a Deterministic sequence. Modulating the least significant bit does not result in human perceptible difference because the amplitude of the change is small. In this technique, the embedding capacity can be increased by using two or more least significant bits. At the same time, not only the risk of making the embedded message statistically detectable increase but also the image fidelity degrades. Hence a variable size LSB embedding schema is presented, in which the number of LSBs used for message embedding/extracting depends on the local characteristics of the pixel. The advantage of LSB-based method is easy to implement and high message pay-load.

Although LSB hides the message in such way that the humans do not perceive it, it is still possible for the opponent to retrieve the message due to the simplicity of the technique. Therefore, malicious people can easily try to extract the message from the beginning of the image if they are suspicious that there exists secret information that was embedded in the image.

Therefore, a system named Secure Information Hiding System (SIHS) is proposed to improve the LSB scheme. It overcomes the sequence-mapping problem by embedding the massage into a set of random pixels, which are scattered on the cover-image.

Masking and filtering techniques, usually restricted to 24 bits and gray scale image, hide information by marking an image, in a manner similar to paper

watermarks. The technique perform analysis of the image, thus embed the information in significant areas so that the hidden message is more integral to cover image than just hiding it in the noise level. Transform techniques embed the message by modulating coefficient in a transform domain, such as the Discrete Fourier Transform, or Wavelet Transform. These methods hide messages in significant areas of the cover image, which make them more robust to attack. Transformations can be applied over the entire image, to block throughout the image, or other variant.

## Image Steganography and bitmap pictures:

Using bitmap pictures for hiding secret information is one of most popular choices for Steganography. Many types of software built for this purpose, some of these software use password protection to encrypting information on picture. To use these software you must have a 'BMP' format of a pictures to use it, but using other type of pictures like "JPEG", "GIF" or any other types is rather or never used, because of algorithm of "BMP" pictures for Steganography is simple. Also we know that in the web most popular of image types are "JPEG" and other types not "BPM", so we should have a solution for this problem. This software provide the solution of this problem, it can accept any type of image to hide information file, but finally it give the only "BMP" image as an output that has hidden file inside it.

## Bitmap Steganography:

Bitmap type is the simplest type of picture because that it doesn't have any technology for decreasing file size. Structure of these files is that a bitmap image created from pixels that any pixel created from three colors (red, green and blue said RGB) each color of a pixel is one byte information that shows the density of that color. Merging these three color makes every color that we see in these pictures. We know that every byte in computer science is created from 8 bit that first bit is Most-Significant-Bit (MSB) and last bit Least-Significant-Bit (LSB), the idea of using Steganography science is in this place; we use LSB bit for writing our security information inside BMP pictures. So if we just use last layer (8st layer) of information, we should change the last bit of pixels, in other hands we have 3 bits in each pixel so we have 3*high*width bits memory to write our information. But before writing our data we must write name of data (file), size of name of data & size of data. We can do this by assigning some first bits of memory (8st layer).

(00101101 00011101 11011100)

(10100110 11000101 00001100)

(11010010 10101100 01100011)

Using each 3 pixel of picture to save a byte of data

# System Analysis & Design

People for long time have tried to sort out the problems faced in the general digital communication system but as these problems exist even now, a secured and easy transfer method has evolved which does the Encryption and Decryption of the file to be sent and hiding this file inside a BMP file. The file to be sent goes through the cryptographic standards and Steganography Techniques. The advantages of this BMP Steganography are:

1. High level Security
2. Cost-effective transfer

SteganoTech – a Steganography system requires any type of image file and the information or message that is to be hidden. It has two modules encrypt and decrypt. Microsoft .Net framework prepares a huge amount of tool and options for programmers that they simples programming. One of .Net tools for pictures and images is auto-converting most types of pictures to BMP format. I used this tool in this software called "SteganoTech" that is written in C# .Net language and you can use this software to hide your information in any type of pictures without any converting its format to BMP (software converts inside it).

The algorithm used for Encryption and Decryption in this application provides using several layers lieu of using only LSB layer of image. Writing data starts from last layer (8st or LSB layer); because significant of this layer is least and every upper layer has doubled significant from its down layer. So every step we go to upper layer image quality decreases and image retouching transpires.

The encrypt module is used to hide information into the image; no one can see that information or file. This module requires any type of image and message and gives the only one image file in destination. The decrypt module is used to get the hidden information in an image file. It take the image file as an output, and give two file at destination folder, one is the same image file and another is the message file that is hidden it that.

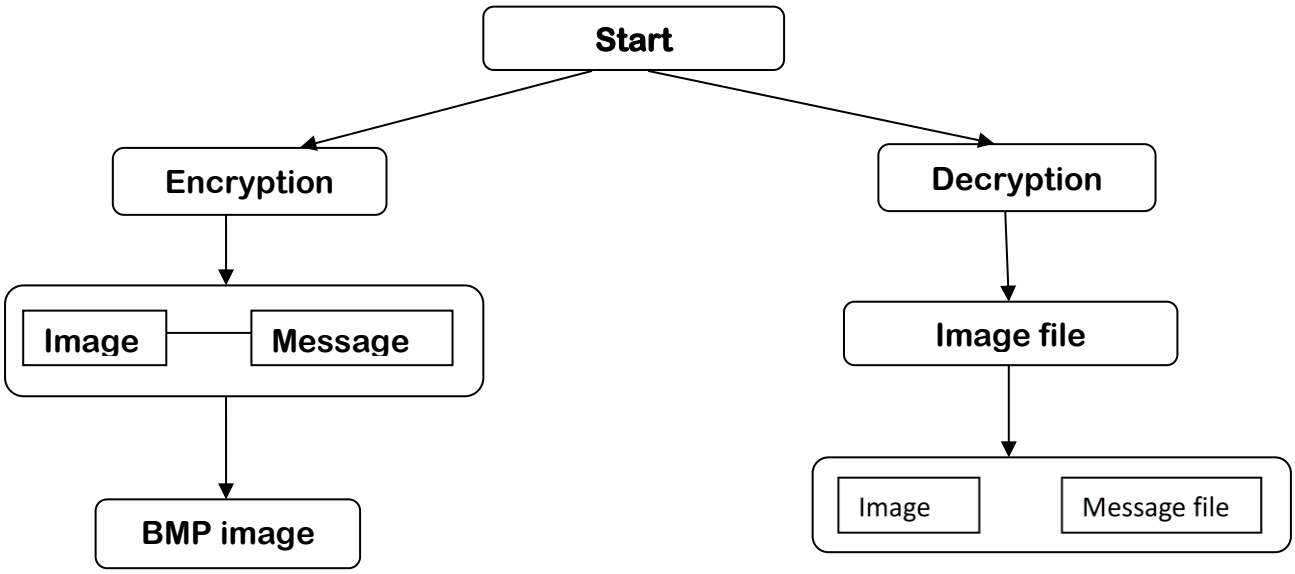
Before encrypting file inside image we must save name and size of file in a definite place of image. We could save file name before file information in LSB layer and save file size and file name size in most right-down pixels of image. Writing this information is needed to retrieve file from encrypted image in decryption state.

In the world where every individual is free to access the information over the network hacking the information over the network has not remained as big task anymore. Many organizations send the information in and out of their network at various levels, which might need security.

If the organizations have this application, then each User can send the information to any other registered User and thus can establish communication and perform the prescribed tasks in secured fashion. The BMP file that the User sends reaches the destinations, within no time in a BMP file format where the end

user needs to de-embed the file, decrypt it and de compress and use for the purpose. The various branches of the organization can be connected to a single host server and then a User of one branch can send files to the User of another branch through the server but in a secured format.

The graphical representation of this system is as follows:




## Operating System:

Windows 10

## Software Requirements:

.NET Framework 4.0 or Greater

## Hardware Requirements:

**Processor** : Preferably 2.0 GHz or Greater.

**RAM** : 4 GB or Greater.

## System Configuration:

**Processor** : Intel(R) Core(TM) i7-6500U CPU @ 2.50GHz (4 CPUs)

**Memory** : 16384 MB RAM

**Hard Disk** : 2 TB HDD

**Network** : 32 Bit PCI Ethernet Card

# Overview

The word steganography comes from the Greek “Seganos”, which mean covered or secret and – “graphy” mean writing or drawing. Therefore, steganography mean, literally, covered writing. It is the art and science of hiding information such its presence cannot be detected and a communication is happening. A secret information is encoding in a manner such that the very existence of the information is concealed. Paired with existing communication methods, steganography can be used to carry out hidden exchanges.

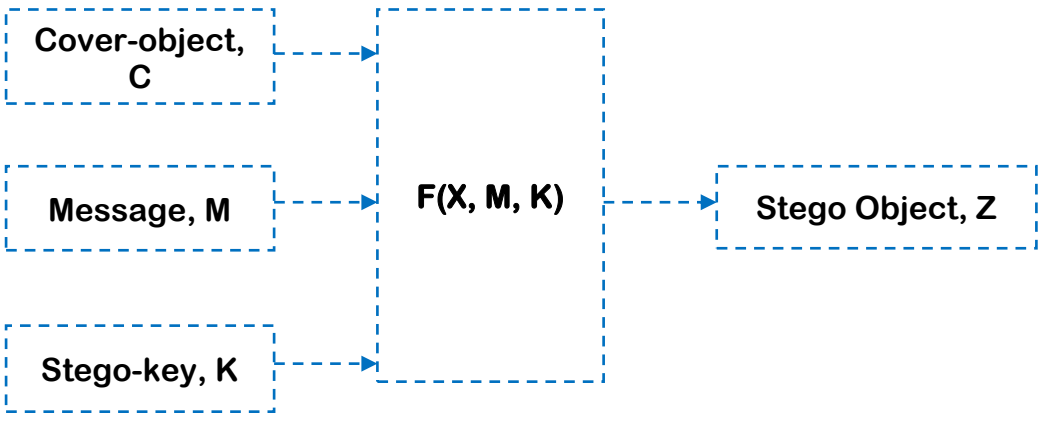
The main goal of this projects it to communicate securely in a completely undetectable manner and to avoid drawing suspicion to the transmission of a hider data. There has been a rapid growth of interest in steganography for two reasons:

1. The publishing and broadcasting industries have become interested in techniques for hiding encrypted copyright marks and serial numbers in digital films, BMP recordings, books and multimedia products

2. Moves by various governments to restrict the availability of encryption services have motivated people to study methods by which private messages can be embedded in seemingly innocuous cover messages.

The basic model of steganography consists of Carrier, Message and password. Carrier is also known as cover-object, which the message is embedded and serves to hide the presence of the message.

Basically, the model for steganography is shown on following figure:



Message is the data that the sender wishes to remain it confidential. It can be plain text, cipher text, other image, or anything that can be embedded in a bit stream such as a copyright mark, a covert communication, or a serial number. Password is known as Stego-key, which ensures that only recipient who know the corresponding decoding key will be able to extract the message from a cover-object. The cover-object with the secretly embedded message is then called the Stego-object.

Recovering message from a Stego-object requires the cover-object itself and a corresponding decoding key if a Stego-key was used during the encoding process. The original image may or may not be required in most applications to extract the message.

There are several suitable carriers below to be the cover-object:

1. Network protocols such as TCP, IP and UDP

2. BMP that using digital BMP formats such as wav, midi, avi, mpeg, mpi and voc

3. File and Disk that can hides and append files by using the slack space

4. Text such as null characters, just alike Morse code including html and .NEt

5. Images file such as bmp, gif and jpg, where they can be both color and gray-scale.

In general, the information hiding process extracts redundant bits from cover-object. The process consists of two steps:

1. Identification of redundant bits in a cover-object. Redundant bits are those bits that can be modified without corrupting the quality or destroying the integrity of the cover-object.

2. Embedding process then selects the subset of the redundant bits to be replaced with data from a secret message. The Stego-object is created by replacing the selected redundant bits with message bits

# Feasibility Study

A feasibility study is a high-level prototype version of the entire System analysis and Design Process. The study begins by classifying the problem definition. Feasibility is to determine if it is worth implementing the set of requirements with approaches mentioned which include the algorithms and the programming language used. Once an acceptance problem definition has been generated, the development of logical model of the system will begin. A search for alternatives is analyzed carefully. There are 3 parts in feasibility study.

1] Operational Feasibility

Question which arise over here are:

- Will the system be used if it developed and implemented?
- If there is sufficient support for the project by the platform on which it is developed?
- Test cases where the system might fail and methods to overcome those problems.
- Will this application be compatible with different platforms?

This system might face a performance issue when trying to embed the data directly into the BMP file but the compression technique addresses that problem which enhances the performance to a great extent. This system being developed in .NEt helps a lot because one of the features of .NEt is platform independence which addresses the compatibility issue.

2] Technical feasibility

- Does the necessary technology exist to do what is been suggested?
- Does the proposed equipment have the technical capacity for using the new system?
- Are there technical guarantees of accuracy, reliability and data security?

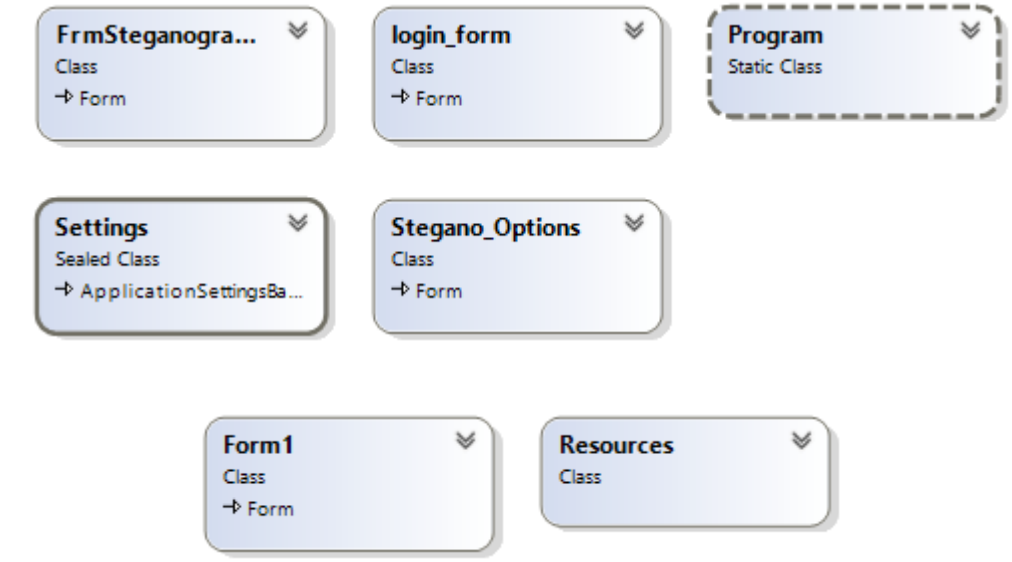
The above mentioned issues are addressed by the following approaches:

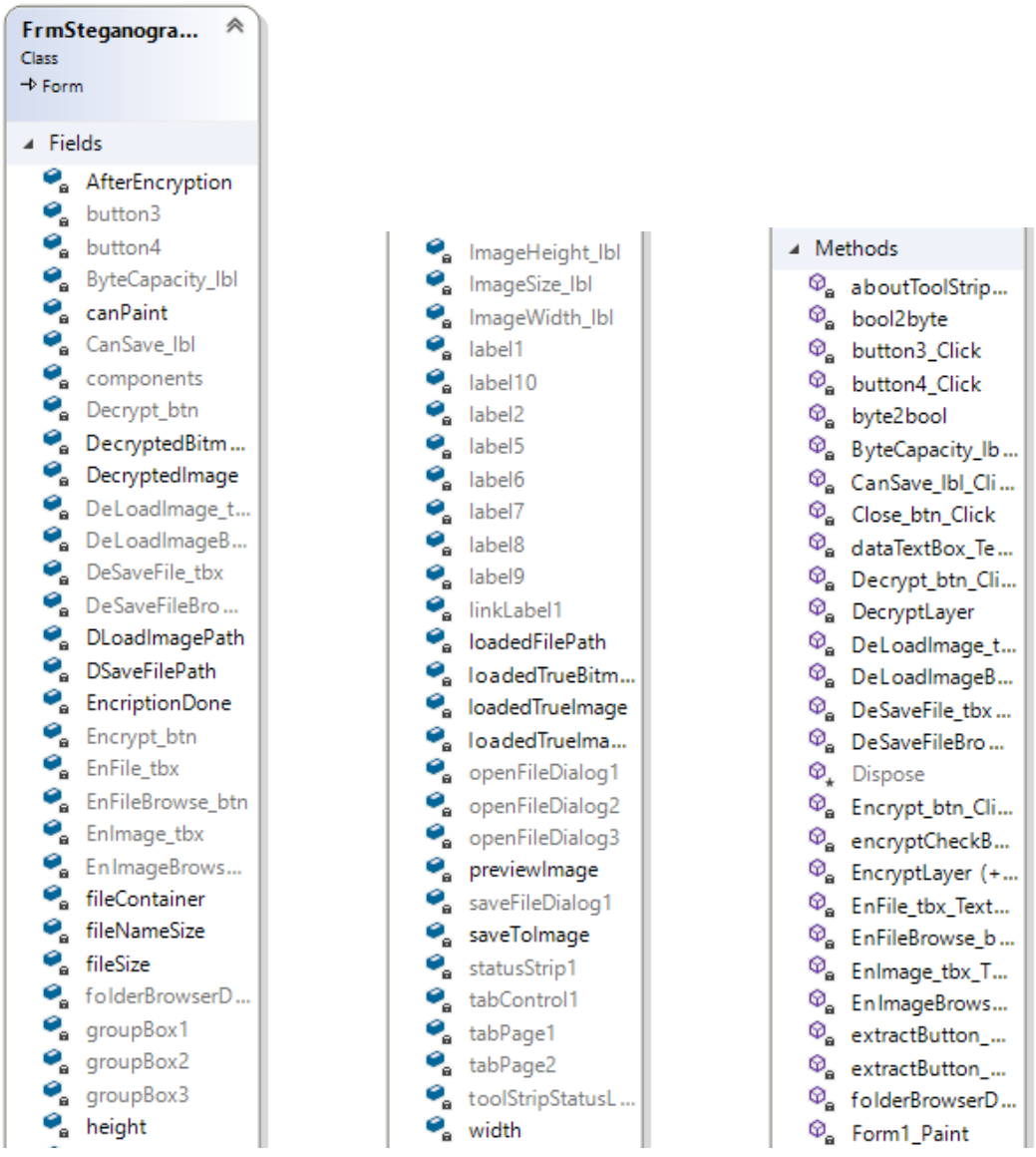
- The environment required in the development of system is any platform.
- The observer pattern along with factory pattern will update the results eventually.
- The language used in the development is .NET & Windows Environment

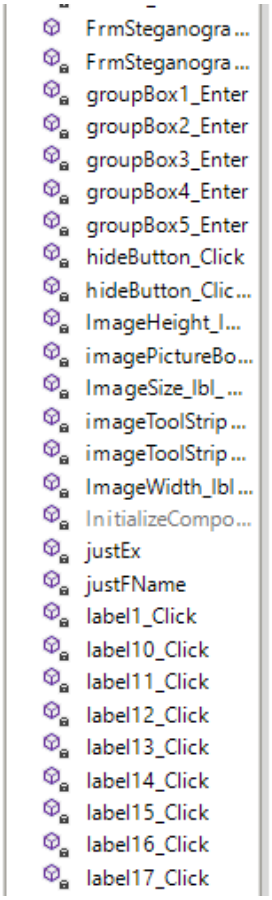
3] Financial and Economic Feasibility

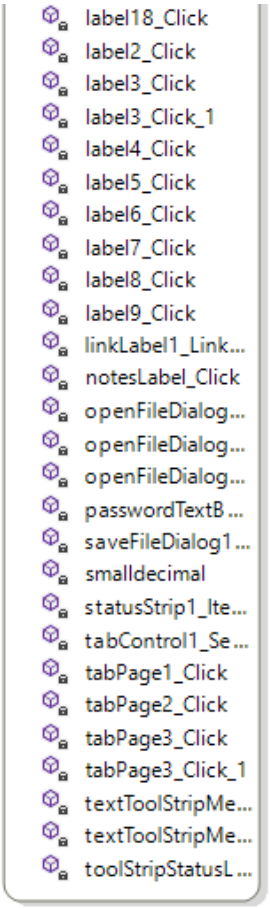
The system developed and installed will be good benefit to the organization. The system will be developed and operated in the existing hardware and software infrastructure. So there is no need of additional hardware and software for the system.

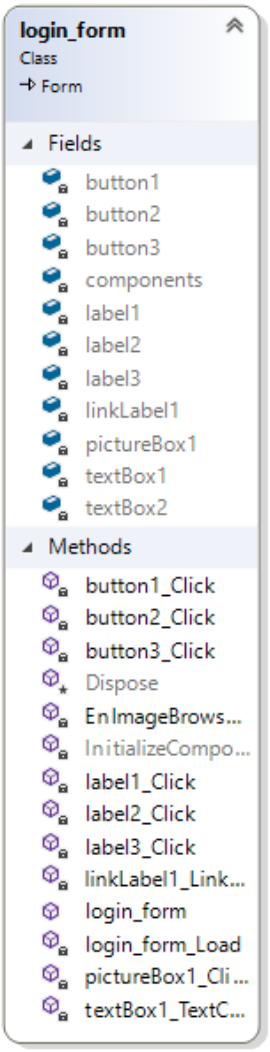
# Class Diagrams

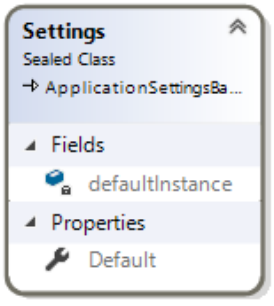


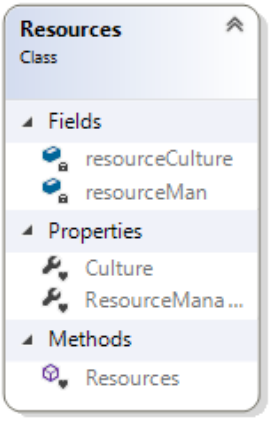



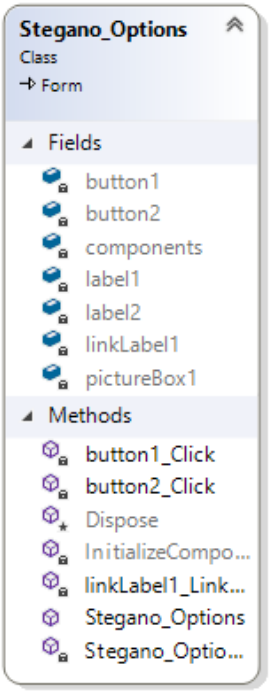

FrmSteganogra...
FrmSteganogra...
groupBox1_Enter
groupBox2_Enter
groupBox3_Enter
groupBox4_Enter
groupBox5_Enter
hideButton_Click
hideButton_Clic...
ImageHeight_l...
imagePictureBo...
ImageSize_lbl_...
imageToolStrip...
imageToolStrip...
ImageWidth_lbl...
InitializeCompo...
justEx
justFName
label1_Click
label10_Click
label11_Click
label12_Click
label13_Click
label14_Click
label15_Click
label16_Click
label17_Click

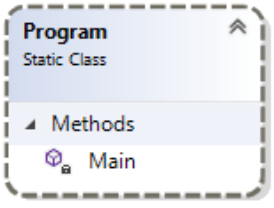

label18_Click
label2_Click
label3_Click
label3_Click_1
label4_Click
label5_Click
label6_Click
label7_Click
label8_Click
label9_Click
linkLabel1_Link...
notesLabel_Click
openFileDialog...
openFileDialog...
openFileDialog...
passwordTextB...
saveFileDialog1...
smalldecimal
statusStrip1_Ite...
tabControl1_Se...
tabPage1_Click
tabPage2_Click
tabPage3_Click
tabPage3_Click_1
textToolStripMe...
textToolStripMe...
toolStripStatusL...

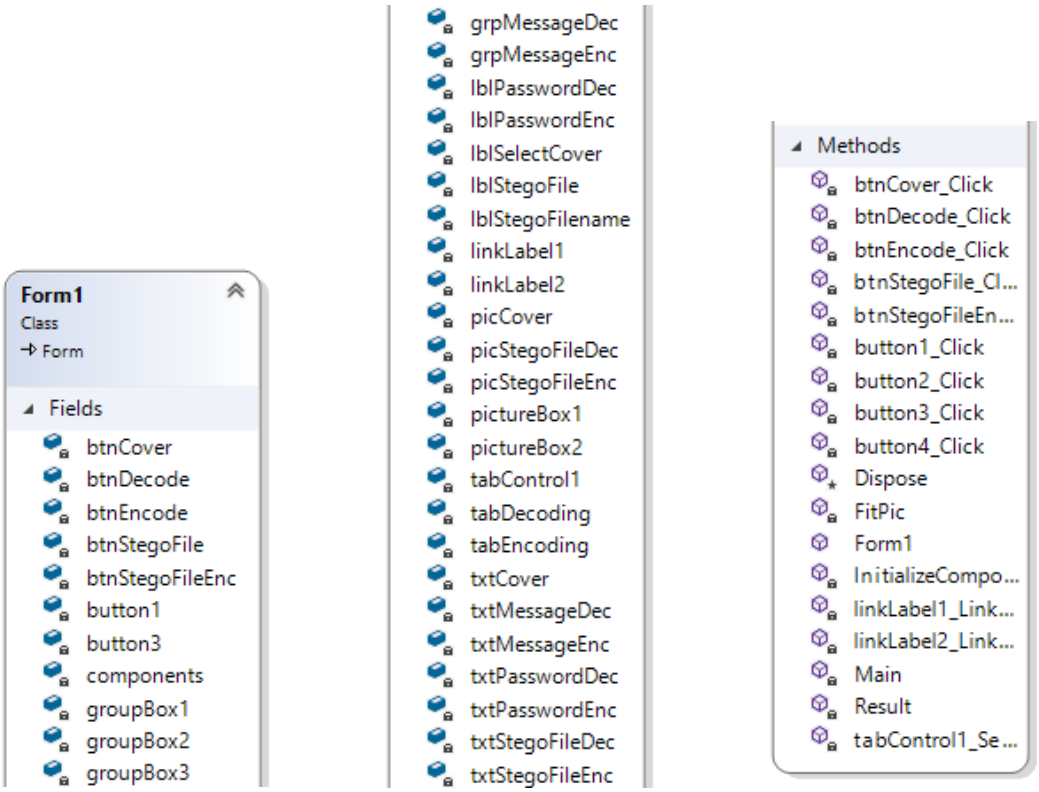

login_form
Class
Form
Fields
button1
button2
button3
components
label1
label2
label3
linkLabel1
pictureBox1
textBox1
textBox2
Methods
button1_Click
button2_Click
button3_Click
Dispose
EnImageBrows...
InitializeCompo...
label1_Click
label2_Click
label3_Click
linkLabel1_Link...
login_form
login_form_Load
pictureBox1_Cli...
textBox1_TextC...


Settings
Sealed Class
ApplicationSettingsBa...
Fields
defaultInstance
Properties
Default


Resources
Class
Fields
resourceCulture
resourceMan
Properties
Culture
ResourceMana...
Methods
Resources

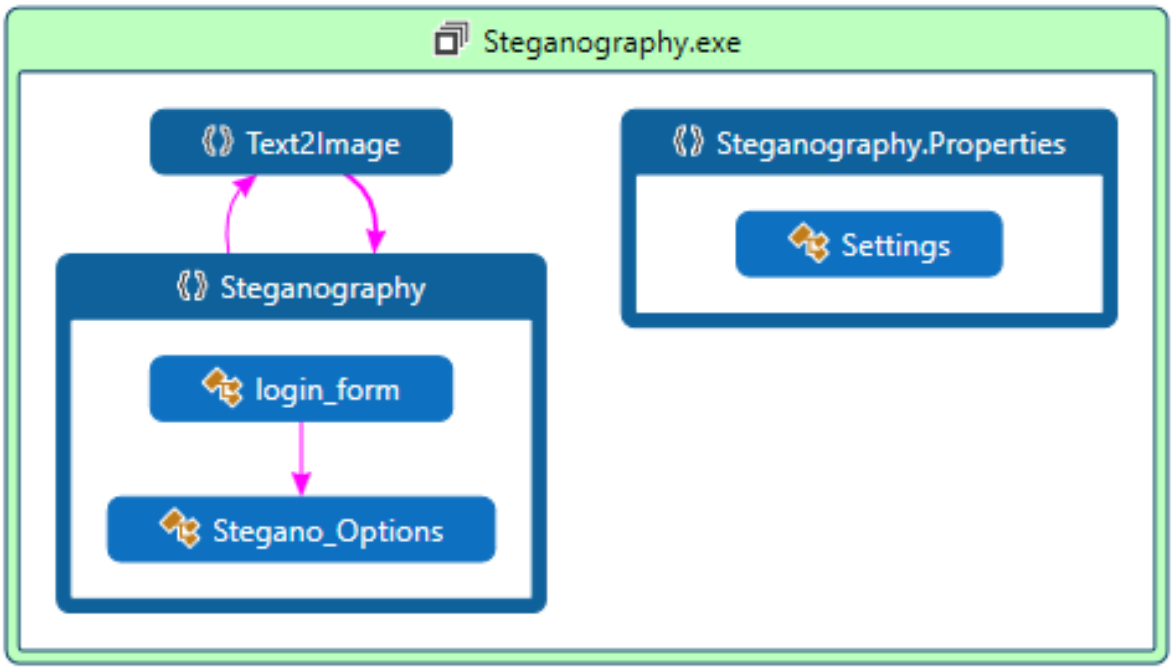

Stegano_Options
Class
Form
Fields
button1
button2
components
label1
label2
linkLabel1
pictureBox1
Methods
button1_Click
button2_Click
Dispose
InitializeCompo...
linkLabel1_Link...
Stegano_Options
Stegano_Optio...


Program
Static Class
Methods
Main

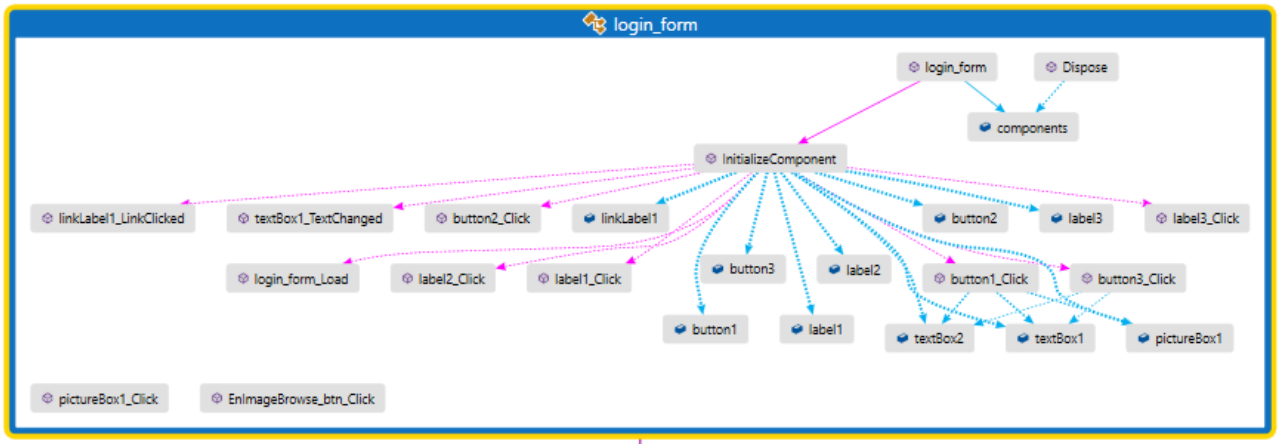



## Site Map

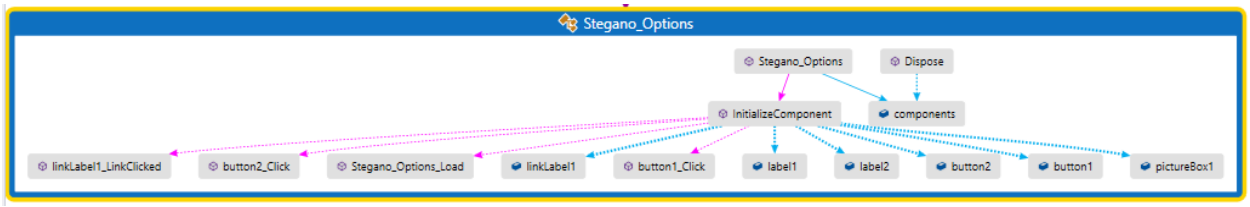
**Steganography_Main**



## Login_Form

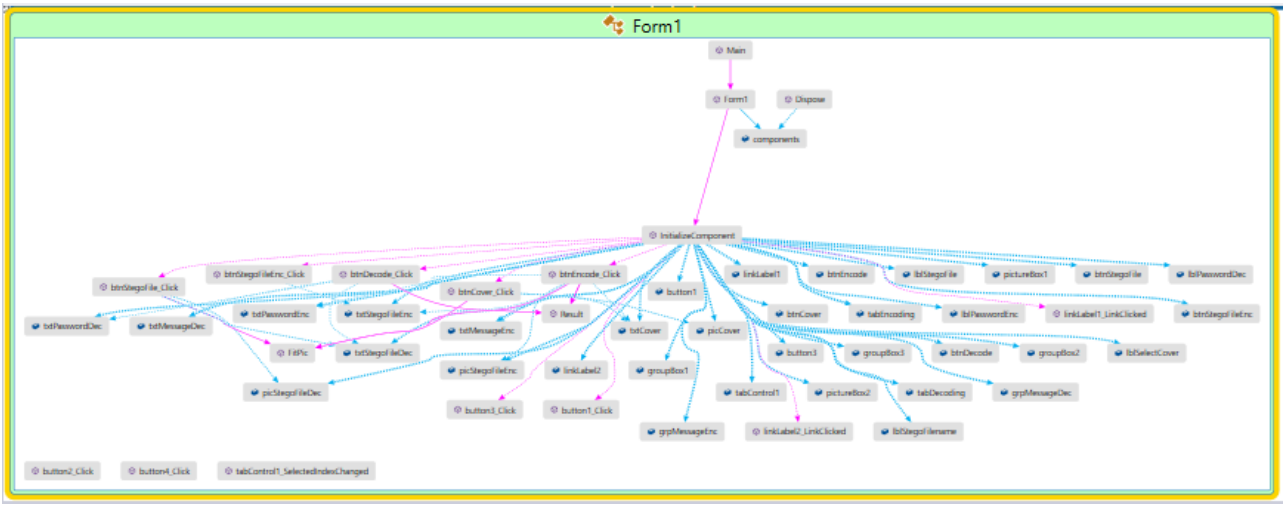



## Stegano_Options



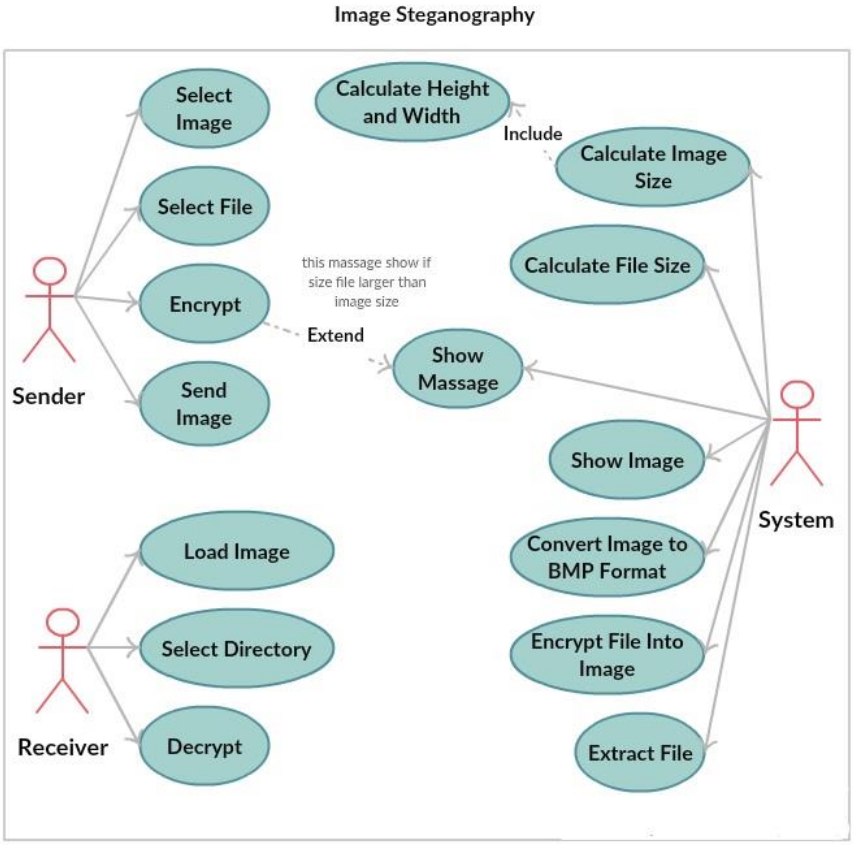

## Image_Steganography

## Text_Steganography

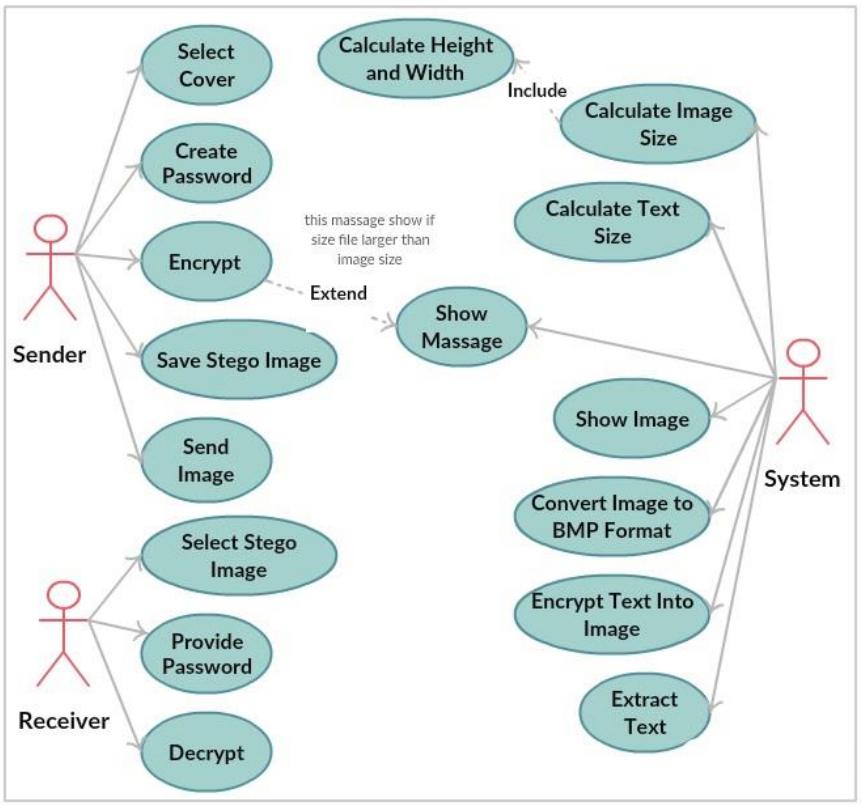


# Use Case Diagrams:

## Image_Steganography

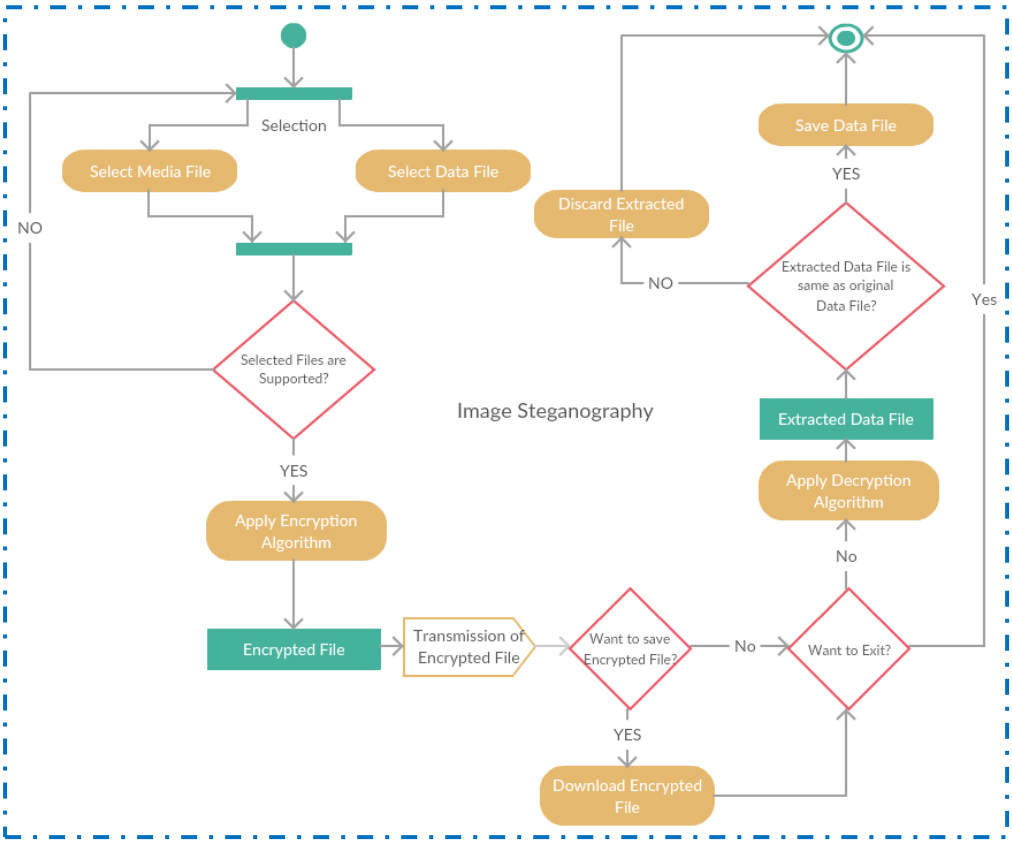



## Text_Steganography

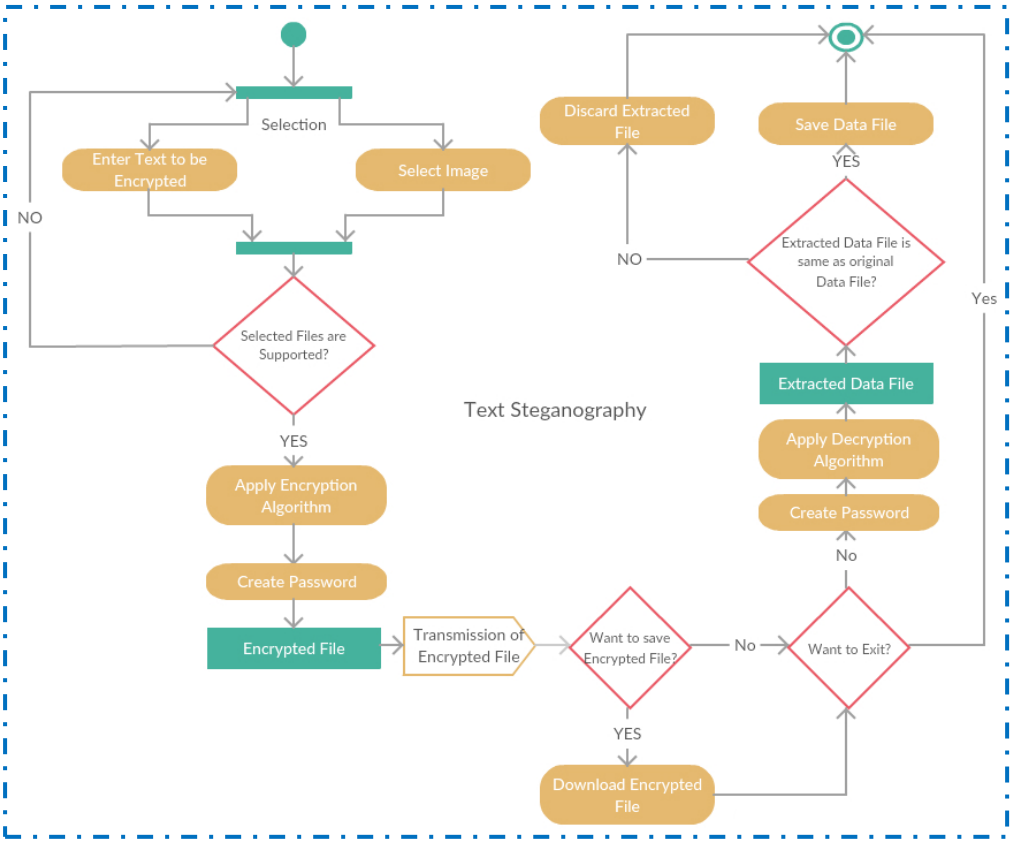


# Activity Flow Diagrams:

## Image_Steganography

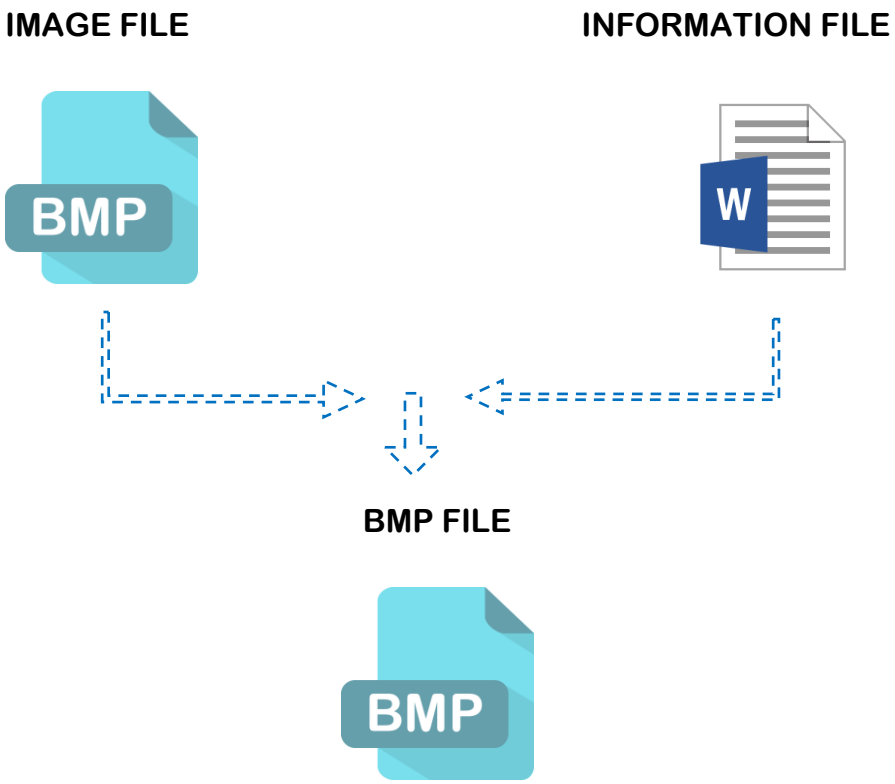



## Text_Steganography

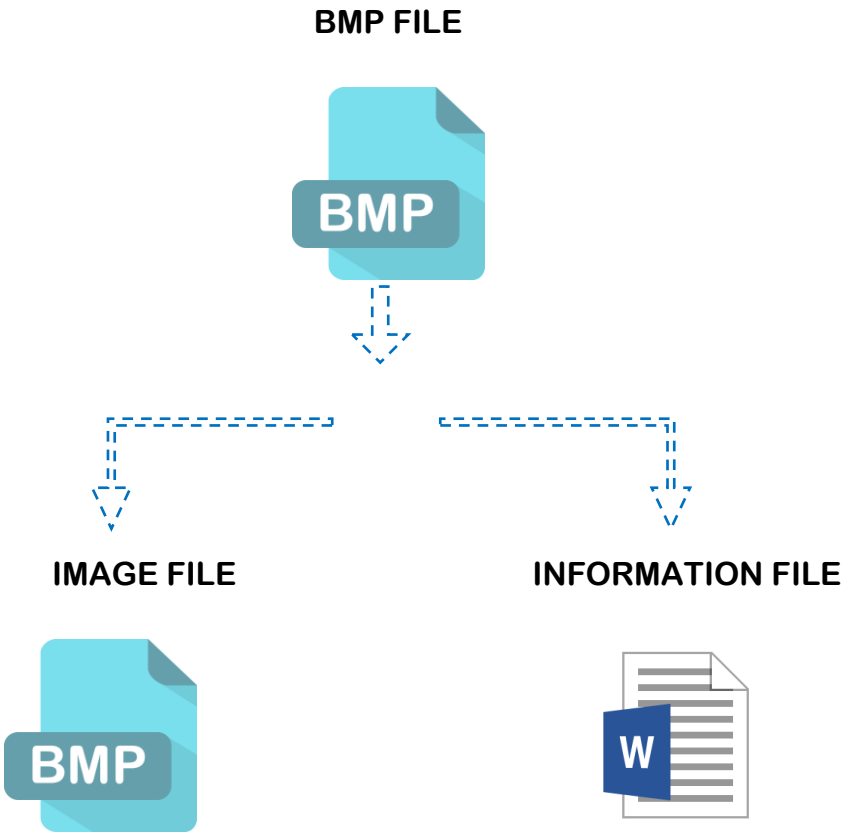


# Image Steganography

## Encryption Process




## Decryption Process

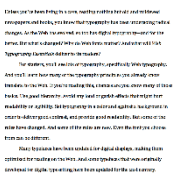


# Text Steganography

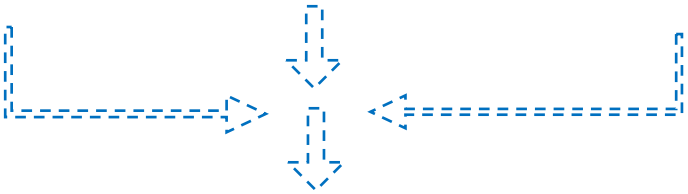
## Encryption Process

IMAGE FILE


INFORMATION TEXT



Create PASSWORD

BMP FILE


## Decryption Process

BMP FILE


Create PASSWORD

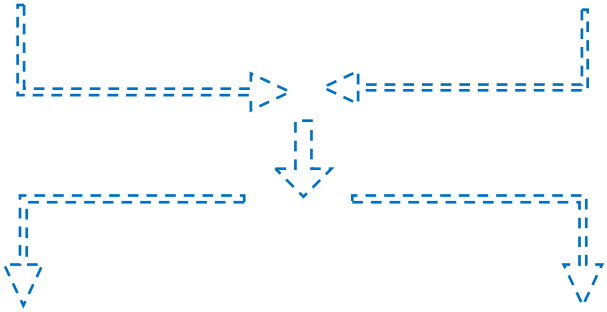
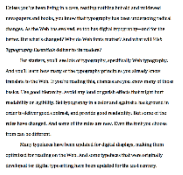



IMAGE FILE

INFORMATION TEXT

# Code Analysis

## Image Steganography: Image_Steganography.cs

```
using System;
using System.Drawing;
using System.Windows.Forms;
using System.IO;

namespace Text2Image
{
    public partial class FrmSteganography : Form
    {
        public FrmSteganography()
        {
            InitializeComponent();
        }

        //public values:
        string loadedTrueImagePath, loadedFilePath,
saveToImage,DLoadImagePath,DSaveFilePath;
        int height, width;
        long fileSize, fileNameSize;
        Image loadedTrueImage, DecryptedImage ,AfterEncryption;
        Bitmap loadedTrueBitmap, DecryptedBitmap;
        Rectangle previewImage = new Rectangle(75,200,475,425);
        bool canPaint = false, EncriptionDone = false;
        byte[] fileContainer;

        private void EnImageBrowse_btn_Click(object sender, EventArgs e)
        {
            if (openFileDialog1.ShowDialog() == DialogResult.OK)
            {
                loadedTrueImagePath = openFileDialog1.FileName;
                EnImage_tbx.Text = loadedTrueImagePath;
                loadedTrueImage = Image.FromFile(loadedTrueImagePath);
                height = loadedTrueImage.Height;
                width = loadedTrueImage.Width;
                loadedTrueBitmap = new Bitmap(loadedTrueImage);

                FileInfo imginf = new FileInfo(loadedTrueImagePath);
                float fs = (float)imginf.Length / 1024;
                ImageSize_lbl.Text = smalldecimal(fs.ToString(), 2) + " KB";
                ImageHeight_lbl.Text = loadedTrueImage.Height.ToString() + "
Pixel";
                ImageWidth_lbl.Text = loadedTrueImage.Width.ToString() + "
Pixel";
                double cansave = (8.0 * ((height * (width / 3) * 3) / 3 - 1)) /
1024;
                CanSave_lbl.Text = smalldecimal(cansave.ToString(), 2) + " KB";

                canPaint = true;
                this.Invalidate();
            }
        }
```

```
        private string smalldecimal(string inp, int dec)
        {
            int i;
            for (i = inp.Length - 1; i > 0; i--)
                if (inp[i] == '.')
                    break;
            try
            {
                return inp.Substring(0, i + dec + 1);
            }
            catch
            {
                return inp;
            }
        }

        private void EnFileBrowse_btn_Click(object sender, EventArgs e)
        {
            if (openFileDialog2.ShowDialog() == DialogResult.OK)
            {
                loadedFilePath = openFileDialog2.FileName;
                EnFile_tbx.Text = loadedFilePath;
                FileInfo finfo = new FileInfo(loadedFilePath);
                fileSize = finfo.Length;
                fileNameSize = justFName(loadedFilePath).Length;
            }
        }

        private void Encrypt_btn_Click(object sender, EventArgs e)
        {
            if (saveFileDialog1.ShowDialog() == DialogResult.OK)
            {
                saveToImage = saveFileDialog1.FileName;
            }
            else
                return;
            if (EnImage_tbx.Text == String.Empty || EnFile_tbx.Text ==
String.Empty)
            {
                MessageBox.Show("Encrypton information is incomplete!\nPlease
complete them frist.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            if (8*((height * (width/3)*3)/3 - 1) < fileSize + fileNameSize)
            {
                MessageBox.Show("File size is too large!\nPlease use a larger
image to hide this file.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
                return;
            }
            fileContainer = File.ReadAllBytes(loadedFilePath);
            EncryptLayer();
        }


        private void EncryptLayer()
        {
            toolStripStatusLabel1.Text ="Encrypting... Please wait";
            Application.DoEvents();
```

```
                long FSize = fileSize;
                Bitmap changedBitmap = EncryptLayer(8, loadedTrueBitmap, 0, (height *
(width/3)*3) / 3 - fileNameSize - 1, true);
                FSize -= (height * (width / 3) * 3) / 3 - fileNameSize - 1;
                if(FSize > 0)
                {
                    for (int i = 7; i >= 0 && FSize > 0; i--)
                    {
                        changedBitmap = EncryptLayer(i, changedBitmap, (((8 - i) *
height * (width / 3) * 3) / 3 - fileNameSize - (8 - i)), (((9 - i) * height *
(width / 3) * 3) / 3 - fileNameSize - (9 - i)), false);
                        FSize -= (height * (width / 3) * 3) / 3 - 1;
                    }
                }
                changedBitmap.Save(saveToImage);
                toolStripStatusLabel1.Text = "Encrypted image has been successfully
saved.";
                MessageBox.Show("Encrypted image has been successfully saved.");
                EncriptionDone = true;
                AfterEncryption = Image.FromFile(saveToImage);
                this.Invalidate();
            }


            private Bitmap EncryptLayer(int layer, Bitmap inputBitmap, long
startPosition, long endPosition, bool writeFileName)
            {
                Bitmap outputBitmap = inputBitmap;
                layer--;
                int i = 0, j = 0;
                long FNSize = 0;
                bool[] t = new bool[8];
                bool[] rb = new bool[8];
                bool[] gb = new bool[8];
                bool[] bb = new bool[8];
                Color pixel = new Color();
                byte r, g, b;

                if (writeFileName)
                {
                    FNSize = fileNameSize;
                    string fileName = justFName(loadedFilePath);

                    //write fileName:
                    for (i = 0; i < height && i * (height / 3) < fileNameSize; i++)
                        for (j = 0; j < (width / 3) * 3 && i * (height / 3) + (j / 3)
< fileNameSize; j++)
                        {
                            byte2bool((byte)fileName[i * (height / 3) + j / 3], ref
t);
                            pixel = inputBitmap.GetPixel(j, i);
                            r = pixel.R;
                            g = pixel.G;
                            b = pixel.B;
                            byte2bool(r, ref rb);
                            byte2bool(g, ref gb);
                            byte2bool(b, ref bb);
                            if (j % 3 == 0)
```

```
                            {
                                rb[7] = t[0];
                                gb[7] = t[1];
                                bb[7] = t[2];
                            }
                            else if (j % 3 == 1)
                            {
                                rb[7] = t[3];
                                gb[7] = t[4];
                                bb[7] = t[5];
                            }
                            else
                            {
                                rb[7] = t[6];
                                gb[7] = t[7];
                            }
                            Color result = Color.FromArgb((int)bool2byte(rb),
(int)bool2byte(gb), (int)bool2byte(bb));
                            outputBitmap.SetPixel(j, i, result);
                        }
                    i--;
                }
                //write file (after file name):
                int tempj = j;

                for (; i < height && i * (height / 3) < endPosition - startPosition +
FNSize && startPosition + i * (height / 3) < fileSize + FNSize; i++)
                    for (j = 0; j < (width / 3) * 3 && i * (height / 3) + (j / 3) <
endPosition - startPosition + FNSize && startPosition + i * (height / 3) + (j /
3) < fileSize + FNSize; j++)
                    {
                        if (tempj != 0)
                        {
                            j = tempj;
                            tempj = 0;
                        }
                        byte2bool((byte)fileContainer[startPosition + i * (height /
3) + j / 3 - FNSize], ref t);
                        pixel = inputBitmap.GetPixel(j, i);
                        r = pixel.R;
                        g = pixel.G;
                        b = pixel.B;
                        byte2bool(r, ref rb);
                        byte2bool(g, ref gb);
                        byte2bool(b, ref bb);
                        if (j % 3 == 0)
                        {
                            rb[layer] = t[0];
                            gb[layer] = t[1];
                            bb[layer] = t[2];
                        }
                        else if (j % 3 == 1)
                        {
                            rb[layer] = t[3];
                            gb[layer] = t[4];
                            bb[layer] = t[5];
                        }
                        else
```

```
                        {
                            rb[layer] = t[6];
                            gb[layer] = t[7];
                        }
                        Color result = Color.FromArgb((int)bool2byte(rb),
(int)bool2byte(gb), (int)bool2byte(bb));
                        outputBitmap.SetPixel(j, i, result);

                    }
                long tempFS = fileSize, tempFNS = fileNameSize;
                r = (byte)(tempFS % 100);
                tempFS /= 100;
                g = (byte)(tempFS % 100);
                tempFS /= 100;
                b = (byte)(tempFS % 100);
                Color flenColor = Color.FromArgb(r,g,b);
                outputBitmap.SetPixel(width - 1, height - 1, flenColor);

                r = (byte)(tempFNS % 100);
                tempFNS /= 100;
                g = (byte)(tempFNS % 100);
                tempFNS /= 100;
                b = (byte)(tempFNS % 100);
                Color fnlenColor = Color.FromArgb(r,g,b);
                outputBitmap.SetPixel(width - 2, height - 1, fnlenColor);

                return outputBitmap;
            }


            private void DecryptLayer()
            {
                toolStripStatusLabel1.Text = "Decrypting... Please wait";
                Application.DoEvents();
                int i, j = 0;
                bool[] t = new bool[8];
                bool[] rb = new bool[8];
                bool[] gb = new bool[8];
                bool[] bb = new bool[8];
                Color pixel = new Color();
                byte r, g, b;
                pixel = DecryptedBitmap.GetPixel(width - 1, height - 1);
                long fSize = pixel.R + pixel.G * 100 + pixel.B * 10000;
                pixel = DecryptedBitmap.GetPixel(width - 2, height - 1);
                long fNameSize = pixel.R + pixel.G * 100 + pixel.B * 10000;
                byte[] res = new byte[fSize];
                string resFName = "";
                byte temp;

                //Read file name:
                for (i = 0; i < height && i * (height / 3) < fNameSize; i++)
                    for (j = 0; j < (width / 3) * 3 && i * (height / 3) + (j / 3) <
fNameSize; j++)
                    {
                        pixel = DecryptedBitmap.GetPixel(j, i);
                        r = pixel.R;
                        g = pixel.G;
                        b = pixel.B;
```

```
                        byte2bool(r, ref rb);
                        byte2bool(g, ref gb);
                        byte2bool(b, ref bb);
                        if (j % 3 == 0)
                        {
                            t[0] = rb[7];
                            t[1] = gb[7];
                            t[2] = bb[7];
                        }
                        else if (j % 3 == 1)
                        {
                            t[3] = rb[7];
                            t[4] = gb[7];
                            t[5] = bb[7];
                        }
                        else
                        {
                            t[6] = rb[7];
                            t[7] = gb[7];
                            temp = bool2byte(t);
                            resFName += (char)temp;
                        }
                    }

                //Read file on layer 8 (after file name):
                int tempj = j;
                i--;

                for (; i < height && i * (height / 3) < fSize + fNameSize; i++)
                    for (j = 0; j < (width / 3) * 3 && i * (height / 3) + (j / 3) <
(height * (width / 3) * 3) / 3 - 1 && i * (height / 3) + (j / 3) < fSize +
fNameSize; j++)
                    {
                        if (tempj != 0)
                        {
                            j = tempj;
                            tempj = 0;
                        }
                        pixel = DecryptedBitmap.GetPixel(j, i);
                        r = pixel.R;
                        g = pixel.G;
                        b = pixel.B;
                        byte2bool(r, ref rb);
                        byte2bool(g, ref gb);
                        byte2bool(b, ref bb);
                        if (j % 3 == 0)
                        {
                            t[0] = rb[7];
                            t[1] = gb[7];
                            t[2] = bb[7];
                        }
                        else if (j % 3 == 1)
                        {
                            t[3] = rb[7];
                            t[4] = gb[7];
                            t[5] = bb[7];
                        }
                        else
```

```
                        {
                            t[6] = rb[7];
                            t[7] = gb[7];
                            temp = bool2byte(t);
                            res[i * (height / 3) + j / 3 - fNameSize] = temp;
                        }
                    }

                //Read file on other layers:
                long readedOnL8 = (height * (width/3)*3) /3 - fNameSize - 1;

                for (int layer = 6; layer >= 0 && readedOnL8 + (6 - layer) * ((height
* (width / 3) * 3) / 3 - 1) < fSize; layer--)
                    for (i = 0; i < height && i * (height / 3) + readedOnL8 + (6 -
layer) * ((height * (width / 3) * 3) / 3 - 1) < fSize; i++)
                        for (j = 0; j < (width / 3) * 3 && i * (height / 3) + (j / 3)
+ readedOnL8 + (6 - layer) * ((height * (width / 3) * 3) / 3 - 1) < fSize; j++)
                        {
                            pixel = DecryptedBitmap.GetPixel(j, i);
                            r = pixel.R;
                            g = pixel.G;
                            b = pixel.B;
                            byte2bool(r, ref rb);
                            byte2bool(g, ref gb);
                            byte2bool(b, ref bb);
                            if (j % 3 == 0)
                            {
                                t[0] = rb[layer];
                                t[1] = gb[layer];
                                t[2] = bb[layer];
                            }
                            else if (j % 3 == 1)
                            {
                                t[3] = rb[layer];
                                t[4] = gb[layer];
                                t[5] = bb[layer];
                            }
                            else
                            {
                                t[6] = rb[layer];
                                t[7] = gb[layer];
                                temp = bool2byte(t);
                                res[i * (height / 3) + j / 3 + (6 - layer) * ((height
* (width / 3) * 3) / 3 - 1) + readedOnL8] = temp;
                            }
                        }

                if (File.Exists(DSaveFilePath + "\\" + resFName))
                {
                    MessageBox.Show("File \"" + resFName + "\" already exist please
choose another path to save file",
"Error",MessageBoxButtons.OK,MessageBoxIcon.Error);
                    return;
                }
                else
                    File.WriteAllBytes(DSaveFilePath + "\\" + resFName, res);
                toolStripStatusLabel1.Text = "Decrypted file has been successfully
saved.";
```

```
            MessageBox.Show("Decrypted file has been successfully saved.");
            Application.DoEvents();
        }

        private void groupBox2_Enter(object sender, EventArgs e)
        {

        }

        private void label3_Click(object sender, EventArgs e)
        {

        }

        private void aboutToolStripMenuItem_Click(object sender, EventArgs e)
        {

        }

        private void imageToolStripMenuItem1_Click(object sender, EventArgs e)
        {

        }

        private void textToolStripMenuItem1_Click(object sender, EventArgs e)
        {

        }

        private void imageToolStripMenuItem_Click(object sender, EventArgs e)
        {

        }

        private void textToolStripMenuItem_Click(object sender, EventArgs e)
        {

        }

        private void extractButton_Click(object sender, EventArgs e)
        {

        }

        private void hideButton_Click(object sender, EventArgs e)
        {

        }

        private void tabPage3_Click(object sender, EventArgs e)
        {

        }

        private void groupBox1_Enter(object sender, EventArgs e)
        {

        }
```

```
        private void label5_Click(object sender, EventArgs e)
        {

        }

        private void DeLoadImage_tbx_TextChanged(object sender, EventArgs e)
        {

        }

        private void DeSaveFile_tbx_TextChanged(object sender, EventArgs e)
        {

        }

        private void label6_Click(object sender, EventArgs e)
        {

        }

        private void label2_Click(object sender, EventArgs e)
        {

        }

        private void EnFile_tbx_TextChanged(object sender, EventArgs e)
        {

        }

        private void label1_Click(object sender, EventArgs e)
        {

        }

        private void EnImage_tbx_TextChanged(object sender, EventArgs e)
        {

        }

        private void openFileDialog1_FileOk(object sender,
    System.ComponentModel.CancelEventArgs e)
        {

        }

        private void openFileDialog2_FileOk(object sender,
    System.ComponentModel.CancelEventArgs e)
        {

        }

        private void saveFileDialog1_FileOk(object sender,
    System.ComponentModel.CancelEventArgs e)
        {

        }
```

```
        private void openFileDialog3_FileOk(object sender,
    System.ComponentModel.CancelEventArgs e)
        {

        }

        private void groupBox3_Enter(object sender, EventArgs e)
        {

        }

        private void ByteCapacity_lbl_Click(object sender, EventArgs e)
        {

        }

        private void CanSave_lbl_Click(object sender, EventArgs e)
        {

        }

        private void ImageWidth_lbl_Click(object sender, EventArgs e)
        {

        }

        private void ImageHeight_lbl_Click(object sender, EventArgs e)
        {

        }

        private void ImageSize_lbl_Click(object sender, EventArgs e)
        {

        }

        private void label9_Click(object sender, EventArgs e)
        {

        }

        private void label10_Click(object sender, EventArgs e)
        {

        }

        private void label8_Click(object sender, EventArgs e)
        {

        }

        private void label7_Click(object sender, EventArgs e)
        {

        }

        private void folderBrowserDialog1_HelpRequest(object sender, EventArgs e)
```

```
        {

        }

        private void statusStrip1_ItemClicked(object sender,
    ToolStripItemClickedEventArgs e)
        {

        }

        private void toolStripStatusLabel1_Click(object sender, EventArgs e)
        {

        }

        private void tabControl1_SelectedIndexChanged(object sender, EventArgs e)
        {

        }

        private void tabPage1_Click(object sender, EventArgs e)
        {

        }

        private void tabPage2_Click(object sender, EventArgs e)
        {

        }

        private void groupBox5_Enter(object sender, EventArgs e)
        {

        }

        private void label3_Click_1(object sender, EventArgs e)
        {

        }

        private void label11_Click(object sender, EventArgs e)
        {

        }

        private void label12_Click(object sender, EventArgs e)
        {

        }

        private void label13_Click(object sender, EventArgs e)
        {

        }

        private void label14_Click(object sender, EventArgs e)
        {
```

```
        }

        private void label15_Click(object sender, EventArgs e)
        {

        }

        private void label16_Click(object sender, EventArgs e)
        {

        }

        private void label17_Click(object sender, EventArgs e)
        {

        }

        private void label18_Click(object sender, EventArgs e)
        {

        }

        private void notesLabel_Click(object sender, EventArgs e)
        {

        }

        private void groupBox4_Enter(object sender, EventArgs e)
        {

        }

        private void label4_Click(object sender, EventArgs e)
        {

        }

        private void passwordTextBox_TextChanged(object sender, EventArgs e)
        {

        }

        private void encryptCheckBox_CheckedChanged(object sender, EventArgs e)
        {

        }

        private void imagePictureBox_Click(object sender, EventArgs e)
        {

        }

        private void extractButton_Click_1(object sender, EventArgs e)
        {

        }

        private void dataTextBox_TextChanged(object sender, EventArgs e)
```

```
        {

        }

        private void hideButton_Click_1(object sender, EventArgs e)
        {

        }

        private void tabPage3_Click_1(object sender, EventArgs e)
        {

        }

        private void FrmSteganography_Load(object sender, EventArgs e)
        {

        }



        private void button4_Click(object sender, EventArgs e)
        {
            Form frm5 = new Steganography.Stegano_Options();

            this.Hide();

            frm5.Closed += (s, args) => this.Close();

            frm5.Show();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            Form frm6 = new Steganography.login_form();

            this.Hide();

            frm6.Closed += (s, args) => this.Close();

            frm6.Show();
        }

        private void byte2bool(byte inp, ref bool[] outp)
        {
            if(inp>=0 && inp<=255)
                for (short i = 7; i >= 0; i--)
                {
                    if (inp % 2 == 1)
                        outp[i] = true;
                    else
                        outp[i] = false;
                    inp /= 2;
                }
            else
                throw new Exception("Input number is illegal.");
        }
```

```
        private byte bool2byte(bool[] inp)
        {
            byte outp = 0;
            for (short i = 7; i >= 0; i--)
            {
                if (inp[i])
                    outp += (byte)Math.Pow(2.0, (double)(7-i));
            }
            return outp;
        }

        private void Decrypt_btn_Click(object sender, EventArgs e)
        {


            if (DeSaveFile_tbx.Text == String.Empty || DeLoadImage_tbx.Text ==
String.Empty)
            {
                MessageBox.Show("Text boxes must not be empty!", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);

                return;
            }

            if (System.IO.File.Exists(DeLoadImage_tbx.Text) == false)
            {
                MessageBox.Show("Select image file.", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                DeLoadImage_tbx.Focus();
                return;
            }



            DecryptLayer();
        }

        private void DeLoadImageBrowse_btn_Click(object sender, EventArgs e)
        {
            if (openFileDialog3.ShowDialog() == DialogResult.OK)
            {
                DLoadImagePath = openFileDialog3.FileName;
                DeLoadImage_tbx.Text = DLoadImagePath;
                DecryptedImage = Image.FromFile(DLoadImagePath);
                height = DecryptedImage.Height;
                width = DecryptedImage.Width;
                DecryptedBitmap = new Bitmap(DecryptedImage);

                FileInfo imginf = new FileInfo(DLoadImagePath);
                float fs = (float)imginf.Length / 1024;
                ImageSize_lbl.Text = smalldecimal(fs.ToString(), 2) + " KB";
                ImageHeight_lbl.Text = DecryptedImage.Height.ToString() + "
Pixel";
                ImageWidth_lbl.Text = DecryptedImage.Width.ToString() + " Pixel";
                double cansave = (8.0 * ((height * (width / 3) * 3) / 3 - 1)) /
1024;
                CanSave_lbl.Text = smalldecimal(cansave.ToString(), 2) + " KB";
```

```
            canPaint = true;
            this.Invalidate();
        }
    }

    private void DeSaveFileBrowse_btn_Click(object sender, EventArgs e)
    {
        if (folderBrowserDialog1.ShowDialog() == DialogResult.OK)
        {
            DSaveFilePath = folderBrowserDialog1.SelectedPath;
            DeSaveFile_tbx.Text = DSaveFilePath;
        }
    }

    private void Form1_Paint(object sender, PaintEventArgs e)
    {
        if(canPaint)
            try
            {
                if (!EncriptionDone)
                    e.Graphics.DrawImage(loadedTrueImage, previewImage);
                else
                    e.Graphics.DrawImage(AfterEncryption, previewImage);
            }
            catch
            {
                e.Graphics.DrawImage(DecryptedImage, previewImage);
            }
    }

    private string justFName(string path)
    {
        string output;
        int i;
        if (path.Length == 3)   // i.e: "C:\\"
            return path.Substring(0, 1);
        for (i = path.Length - 1; i > 0; i--)
            if (path[i] == '\\')
                break;
        output = path.Substring(i + 1);
        return output;
    }


    private string justEx(string fName)
    {
        string output;
        int i;
        for (i = fName.Length - 1; i > 0; i--)
            if (fName[i] == '.')
                break;
        output = fName.Substring(i + 1);
        return output;
    }

    private void Close_btn_Click(object sender, EventArgs e)
    {
        this.Close();
```

```
        }

        private void linkLabel1_LinkClicked(object sender,
LinkLabelLinkClickedEventArgs e)
        {

System.Diagnostics.Process.Start("https://plus.google.com/u/0/104671997220884711082");

        }

    }
}
```

## Text Steganography: Text_Steganography.cs

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using Steganography;

namespace SteganographyGUI
{
      /// <summary>
      /// Summary description for Form1.
      /// </summary>
      public class Form1 : System.Windows.Forms.Form
      {
            private System.Windows.Forms.TabControl tabControl1;
            private System.Windows.Forms.TabPage tabEncoding;
            private System.Windows.Forms.TabPage tabDecoding;
            private System.Windows.Forms.GroupBox groupBox1;
            private System.Windows.Forms.GroupBox groupBox2;
            private System.Windows.Forms.PictureBox picCover;
            private System.Windows.Forms.Label lblSelectCover;
            private System.Windows.Forms.TextBox txtCover;
            private System.Windows.Forms.Button btnCover;
            private System.Windows.Forms.Button btnEncode;
            private System.Windows.Forms.GroupBox groupBox3;
            private System.Windows.Forms.PictureBox picStegoFileEnc;
            private System.Windows.Forms.GroupBox grpMessageEnc;
            private System.Windows.Forms.TextBox txtMessageEnc;
            private System.Windows.Forms.Label lblPasswordEnc;
            private System.Windows.Forms.TextBox txtPasswordEnc;
            private System.Windows.Forms.PictureBox picStegoFileDec;
            private System.Windows.Forms.Label lblStegoFile;
            private System.Windows.Forms.Button btnStegoFile;
            private System.Windows.Forms.Label lblPasswordDec;
            private System.Windows.Forms.TextBox txtPasswordDec;
            private System.Windows.Forms.Button btnDecode;
            private System.Windows.Forms.GroupBox grpMessageDec;
            private System.Windows.Forms.TextBox txtMessageDec;
            private System.Windows.Forms.Label lblStegoFilename;
            private System.Windows.Forms.TextBox txtStegoFileDec;
```

```
            private System.Windows.Forms.TextBox txtStegoFileEnc;
            private System.Windows.Forms.Button btnStegoFileEnc;
            private PictureBox pictureBox1;
            private PictureBox pictureBox2;
            private Button button3;
            private Button button1;
            private LinkLabel linkLabel2;
            private LinkLabel linkLabel1;

        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.Container components = null;

            public Form1()
            {
                  //
                  // Required for Windows Form Designer support
                  //
                  InitializeComponent();

                  //
                  // TODO: Add any constructor code after InitializeComponent
call
                  //
            }

            /// <summary>
            /// Clean up any resources being used.
            /// </summary>
            protected override void Dispose( bool disposing )
            {
                  if( disposing )
                  {
                        if (components != null)
                        {
                              components.Dispose();
                        }
                  }
                  base.Dispose( disposing );
            }

            #region Windows Form Designer generated code
            /// <summary>
            /// Required method for Designer support - do not modify
            /// the contents of this method with the code editor.
            /// </summary>
            private void InitializeComponent()
            {
            System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(Form1));
            this.tabControl1 = new System.Windows.Forms.TabControl();
            this.tabEncoding = new System.Windows.Forms.TabPage();
            this.linkLabel2 = new System.Windows.Forms.LinkLabel();
            this.button3 = new System.Windows.Forms.Button();
            this.pictureBox1 = new System.Windows.Forms.PictureBox();
            this.btnStegoFileEnc = new System.Windows.Forms.Button();
            this.txtStegoFileEnc = new System.Windows.Forms.TextBox();
```

```
                this.lblStegoFilename = new System.Windows.Forms.Label();
                this.btnEncode = new System.Windows.Forms.Button();
                this.txtPasswordEnc = new System.Windows.Forms.TextBox();
                this.lblPasswordEnc = new System.Windows.Forms.Label();
                this.grpMessageEnc = new System.Windows.Forms.GroupBox();
                this.txtMessageEnc = new System.Windows.Forms.TextBox();
                this.btnCover = new System.Windows.Forms.Button();
                this.txtCover = new System.Windows.Forms.TextBox();
                this.lblSelectCover = new System.Windows.Forms.Label();
                this.groupBox2 = new System.Windows.Forms.GroupBox();
                this.picStegoFileEnc = new System.Windows.Forms.PictureBox();
                this.groupBox1 = new System.Windows.Forms.GroupBox();
                this.picCover = new System.Windows.Forms.PictureBox();
                this.tabDecoding = new System.Windows.Forms.TabPage();
                this.linkLabel1 = new System.Windows.Forms.LinkLabel();
                this.button1 = new System.Windows.Forms.Button();
                this.pictureBox2 = new System.Windows.Forms.PictureBox();
                this.grpMessageDec = new System.Windows.Forms.GroupBox();
                this.txtMessageDec = new System.Windows.Forms.TextBox();
                this.btnDecode = new System.Windows.Forms.Button();
                this.txtPasswordDec = new System.Windows.Forms.TextBox();
                this.lblPasswordDec = new System.Windows.Forms.Label();
                this.btnStegoFile = new System.Windows.Forms.Button();
                this.txtStegoFileDec = new System.Windows.Forms.TextBox();
                this.lblStegoFile = new System.Windows.Forms.Label();
                this.groupBox3 = new System.Windows.Forms.GroupBox();
                this.picStegoFileDec = new System.Windows.Forms.PictureBox();
                this.tabControl1.SuspendLayout();
                this.tabEncoding.SuspendLayout();

        ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).BeginInit();
                this.grpMessageEnc.SuspendLayout();
                this.groupBox2.SuspendLayout();

        ((System.ComponentModel.ISupportInitialize)(this.picStegoFileEnc)).BeginInit();
                this.groupBox1.SuspendLayout();

        ((System.ComponentModel.ISupportInitialize)(this.picCover)).BeginInit();
                this.tabDecoding.SuspendLayout();

        ((System.ComponentModel.ISupportInitialize)(this.pictureBox2)).BeginInit();
                this.grpMessageDec.SuspendLayout();
                this.groupBox3.SuspendLayout();

        ((System.ComponentModel.ISupportInitialize)(this.picStegoFileDec)).BeginInit();
                this.SuspendLayout();
                //
                // tabControl1
                //
                this.tabControl1.Controls.Add(this.tabEncoding);
                this.tabControl1.Controls.Add(this.tabDecoding);
                this.tabControl1.Font = new System.Drawing.Font("Comic Sans MS",
        9.75F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
        ((byte)(0)));
                this.tabControl1.Location = new System.Drawing.Point(3, 1);
                this.tabControl1.Name = "tabControl1";
                this.tabControl1.SelectedIndex = 0;
                this.tabControl1.Size = new System.Drawing.Size(615, 538);
```

```
            this.tabControl1.TabIndex = 2;
            //
            // tabEncoding
            //
            this.tabEncoding.BackColor = System.Drawing.Color.White;
            this.tabEncoding.Controls.Add(this.linkLabel2);
            this.tabEncoding.Controls.Add(this.button3);
            this.tabEncoding.Controls.Add(this.pictureBox1);
            this.tabEncoding.Controls.Add(this.btnStegoFileEnc);
            this.tabEncoding.Controls.Add(this.txtStegoFileEnc);
            this.tabEncoding.Controls.Add(this.lblStegoFilename);
            this.tabEncoding.Controls.Add(this.btnEncode);
            this.tabEncoding.Controls.Add(this.txtPasswordEnc);
            this.tabEncoding.Controls.Add(this.lblPasswordEnc);
            this.tabEncoding.Controls.Add(this.grpMessageEnc);
            this.tabEncoding.Controls.Add(this.btnCover);
            this.tabEncoding.Controls.Add(this.txtCover);
            this.tabEncoding.Controls.Add(this.lblSelectCover);
            this.tabEncoding.Controls.Add(this.groupBox2);
            this.tabEncoding.Controls.Add(this.groupBox1);
            this.tabEncoding.Location = new System.Drawing.Point(4, 27);
            this.tabEncoding.Name = "tabEncoding";
            this.tabEncoding.Size = new System.Drawing.Size(607, 507);
            this.tabEncoding.TabIndex = 0;
            this.tabEncoding.Text = "Encoding";
            //
            // linkLabel2
            //
            this.linkLabel2.Anchor =
    ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Bottom |
    System.Windows.Forms.AnchorStyles.Right)));
            this.linkLabel2.AutoSize = true;
            this.linkLabel2.BackColor = System.Drawing.Color.White;
            this.linkLabel2.ForeColor = System.Drawing.Color.Indigo;
            this.linkLabel2.Location = new System.Drawing.Point(3, 3);
            this.linkLabel2.Name = "linkLabel2";
            this.linkLabel2.Size = new System.Drawing.Size(165, 18);
            this.linkLabel2.TabIndex = 33;
            this.linkLabel2.TabStop = true;
            this.linkLabel2.Text = "Copyright © Pawan Gosavi";
            this.linkLabel2.TextAlign =
    System.Drawing.ContentAlignment.BottomRight;
            this.linkLabel2.LinkClicked += new
    System.Windows.Forms.LinkLabelLinkClickedEventHandler(this.linkLabel2_LinkClicked
    );
            //
            // button3
            //
            this.button3.BackColor = System.Drawing.Color.White;
            this.button3.Font = new System.Drawing.Font("Comic Sans MS", 8.25F,
    System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.button3.ForeColor = System.Drawing.Color.Indigo;
            this.button3.Location = new System.Drawing.Point(528, 3);
            this.button3.Name = "button3";
            this.button3.Size = new System.Drawing.Size(75, 26);
            this.button3.TabIndex = 32;
            this.button3.Text = "Exit";
            this.button3.UseVisualStyleBackColor = false;
```

```
            this.button3.Click += new System.EventHandler(this.button3_Click);
            //
            // pictureBox1
            //
            this.pictureBox1.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox1.Image")));
            this.pictureBox1.Location = new System.Drawing.Point(192, 83);
            this.pictureBox1.Name = "pictureBox1";
            this.pictureBox1.Size = new System.Drawing.Size(217, 110);
            this.pictureBox1.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.StretchImage;
            this.pictureBox1.TabIndex = 26;
            this.pictureBox1.TabStop = false;
            //
            // btnStegoFileEnc
            //
            this.btnStegoFileEnc.BackColor = System.Drawing.Color.White;
            this.btnStegoFileEnc.Font = new System.Drawing.Font("Comic Sans MS",
9.75F, ((System.Drawing.FontStyle)((System.Drawing.FontStyle.Bold |
System.Drawing.FontStyle.Italic))), System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.btnStegoFileEnc.ForeColor = System.Drawing.Color.Indigo;
            this.btnStegoFileEnc.Location = new System.Drawing.Point(508, 425);
            this.btnStegoFileEnc.Name = "btnStegoFileEnc";
            this.btnStegoFileEnc.Size = new System.Drawing.Size(75, 30);
            this.btnStegoFileEnc.TabIndex = 19;
            this.btnStegoFileEnc.Text = "Browse";
            this.btnStegoFileEnc.UseVisualStyleBackColor = false;
            this.btnStegoFileEnc.Click += new
System.EventHandler(this.btnStegoFileEnc_Click);
            //
            // txtStegoFileEnc
            //
            this.txtStegoFileEnc.ForeColor = System.Drawing.Color.Red;
            this.txtStegoFileEnc.Location = new System.Drawing.Point(155, 429);
            this.txtStegoFileEnc.Name = "txtStegoFileEnc";
            this.txtStegoFileEnc.Size = new System.Drawing.Size(328, 26);
            this.txtStegoFileEnc.TabIndex = 23;
            //
            // lblStegoFilename
            //
            this.lblStegoFilename.Font = new System.Drawing.Font("Comic Sans MS",
9.75F, ((System.Drawing.FontStyle)((System.Drawing.FontStyle.Bold |
System.Drawing.FontStyle.Italic))), System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.lblStegoFilename.ForeColor = System.Drawing.Color.Indigo;
            this.lblStegoFilename.Location = new System.Drawing.Point(14, 425);
            this.lblStegoFilename.Name = "lblStegoFilename";
            this.lblStegoFilename.Size = new System.Drawing.Size(135, 30);
            this.lblStegoFilename.TabIndex = 17;
            this.lblStegoFilename.Text = "Select Stego File :";
            this.lblStegoFilename.TextAlign =
System.Drawing.ContentAlignment.MiddleCenter;
            //
            // btnEncode
            //
            this.btnEncode.BackColor = System.Drawing.Color.White;
```

```
            this.btnEncode.Font = new System.Drawing.Font("Comic Sans MS", 9.75F,
    ((System.Drawing.FontStyle)((System.Drawing.FontStyle.Bold |
    System.Drawing.FontStyle.Italic))), System.Drawing.GraphicsUnit.Point,
    ((byte)(0)));
            this.btnEncode.ForeColor = System.Drawing.Color.Indigo;
            this.btnEncode.Location = new System.Drawing.Point(508, 464);
            this.btnEncode.Name = "btnEncode";
            this.btnEncode.Size = new System.Drawing.Size(75, 30);
            this.btnEncode.TabIndex = 25;
            this.btnEncode.Text = "Encrypt";
            this.btnEncode.UseVisualStyleBackColor = false;
            this.btnEncode.Click += new
    System.EventHandler(this.btnEncode_Click);
            //
            // txtPasswordEnc
            //
            this.txtPasswordEnc.ForeColor = System.Drawing.Color.Red;
            this.txtPasswordEnc.Location = new System.Drawing.Point(155, 468);
            this.txtPasswordEnc.Name = "txtPasswordEnc";
            this.txtPasswordEnc.PasswordChar = '*';
            this.txtPasswordEnc.Size = new System.Drawing.Size(328, 26);
            this.txtPasswordEnc.TabIndex = 24;
            //
            // lblPasswordEnc
            //
            this.lblPasswordEnc.Font = new System.Drawing.Font("Comic Sans MS",
    9.75F, ((System.Drawing.FontStyle)((System.Drawing.FontStyle.Bold |
    System.Drawing.FontStyle.Italic))), System.Drawing.GraphicsUnit.Point,
    ((byte)(0)));
            this.lblPasswordEnc.ForeColor = System.Drawing.Color.Indigo;
            this.lblPasswordEnc.Location = new System.Drawing.Point(25, 464);
            this.lblPasswordEnc.Name = "lblPasswordEnc";
            this.lblPasswordEnc.Size = new System.Drawing.Size(124, 30);
            this.lblPasswordEnc.TabIndex = 14;
            this.lblPasswordEnc.Text = "Create Password :";
            this.lblPasswordEnc.TextAlign =
    System.Drawing.ContentAlignment.MiddleCenter;
            //
            // grpMessageEnc
            //
            this.grpMessageEnc.Controls.Add(this.txtMessageEnc);
            this.grpMessageEnc.Font = new System.Drawing.Font("Comic Sans MS",
    9.75F, ((System.Drawing.FontStyle)((System.Drawing.FontStyle.Bold |
    System.Drawing.FontStyle.Italic))), System.Drawing.GraphicsUnit.Point,
    ((byte)(0)));
            this.grpMessageEnc.ForeColor = System.Drawing.Color.Indigo;
            this.grpMessageEnc.Location = new System.Drawing.Point(33, 283);
            this.grpMessageEnc.Name = "grpMessageEnc";
            this.grpMessageEnc.Size = new System.Drawing.Size(550, 136);
            this.grpMessageEnc.TabIndex = 13;
            this.grpMessageEnc.TabStop = false;
            this.grpMessageEnc.Text = " Message to hide";
            //
            // txtMessageEnc
            //
            this.txtMessageEnc.Font = new System.Drawing.Font("Comic Sans MS",
    8.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
    ((byte)(0)));
```

```
            this.txtMessageEnc.ForeColor = System.Drawing.Color.Red;
            this.txtMessageEnc.Location = new System.Drawing.Point(6, 25);
            this.txtMessageEnc.Multiline = true;
            this.txtMessageEnc.Name = "txtMessageEnc";
            this.txtMessageEnc.ScrollBars =
System.Windows.Forms.ScrollBars.Vertical;
            this.txtMessageEnc.Size = new System.Drawing.Size(538, 105);
            this.txtMessageEnc.TabIndex = 4;
            //
            // btnCover
            //
            this.btnCover.BackColor = System.Drawing.Color.White;
            this.btnCover.Font = new System.Drawing.Font("Comic Sans MS", 9.75F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.btnCover.ForeColor = System.Drawing.Color.Indigo;
            this.btnCover.Location = new System.Drawing.Point(508, 229);
            this.btnCover.Name = "btnCover";
            this.btnCover.Size = new System.Drawing.Size(75, 30);
            this.btnCover.TabIndex = 21;
            this.btnCover.Text = "Browse";
            this.btnCover.UseVisualStyleBackColor = false;
            this.btnCover.Click += new System.EventHandler(this.btnCover_Click);
            //
            // txtCover
            //
            this.txtCover.ForeColor = System.Drawing.Color.Red;
            this.txtCover.Location = new System.Drawing.Point(155, 233);
            this.txtCover.Name = "txtCover";
            this.txtCover.Size = new System.Drawing.Size(328, 26);
            this.txtCover.TabIndex = 22;
            //
            // lblSelectCover
            //
            this.lblSelectCover.Font = new System.Drawing.Font("Comic Sans MS",
9.75F, ((System.Drawing.FontStyle)((System.Drawing.FontStyle.Bold |
System.Drawing.FontStyle.Italic))), System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.lblSelectCover.ForeColor = System.Drawing.Color.Indigo;
            this.lblSelectCover.Location = new System.Drawing.Point(14, 229);
            this.lblSelectCover.Name = "lblSelectCover";
            this.lblSelectCover.Size = new System.Drawing.Size(110, 30);
            this.lblSelectCover.TabIndex = 9;
            this.lblSelectCover.Text = " Select Cover :";
            this.lblSelectCover.TextAlign =
System.Drawing.ContentAlignment.MiddleCenter;
            //
            // groupBox2
            //
            this.groupBox2.Controls.Add(this.picStegoFileEnc);
            this.groupBox2.Font = new System.Drawing.Font("Comic Sans MS", 9.75F,
((System.Drawing.FontStyle)((System.Drawing.FontStyle.Bold |
System.Drawing.FontStyle.Italic))), System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.groupBox2.ForeColor = System.Drawing.Color.Indigo;
            this.groupBox2.Location = new System.Drawing.Point(415, 39);
            this.groupBox2.Name = "groupBox2";
            this.groupBox2.Size = new System.Drawing.Size(168, 184);
            this.groupBox2.TabIndex = 1;
```

```
                this.groupBox2.TabStop = false;
                this.groupBox2.Text = "Stego File";
                //
                // picStegoFileEnc
                //
                this.picStegoFileEnc.BackColor = System.Drawing.Color.White;
                this.picStegoFileEnc.Location = new System.Drawing.Point(8, 24);
                this.picStegoFileEnc.Name = "picStegoFileEnc";
                this.picStegoFileEnc.Size = new System.Drawing.Size(150, 150);
                this.picStegoFileEnc.SizeMode =
        System.Windows.Forms.PictureBoxSizeMode.StretchImage;
                this.picStegoFileEnc.TabIndex = 2;
                this.picStegoFileEnc.TabStop = false;
                //
                // groupBox1
                //
                this.groupBox1.Controls.Add(this.picCover);
                this.groupBox1.Font = new System.Drawing.Font("Comic Sans MS", 9.75F,
        ((System.Drawing.FontStyle)((System.Drawing.FontStyle.Bold |
        System.Drawing.FontStyle.Italic))), System.Drawing.GraphicsUnit.Point,
        ((byte)(0)));
                this.groupBox1.ForeColor = System.Drawing.Color.Indigo;
                this.groupBox1.Location = new System.Drawing.Point(18, 42);
                this.groupBox1.Name = "groupBox1";
                this.groupBox1.Size = new System.Drawing.Size(168, 184);
                this.groupBox1.TabIndex = 0;
                this.groupBox1.TabStop = false;
                this.groupBox1.Text = "Cover";
                //
                // picCover
                //
                this.picCover.BackColor = System.Drawing.Color.White;
                this.picCover.Location = new System.Drawing.Point(8, 24);
                this.picCover.Name = "picCover";
                this.picCover.Size = new System.Drawing.Size(150, 150);
                this.picCover.SizeMode =
        System.Windows.Forms.PictureBoxSizeMode.StretchImage;
                this.picCover.TabIndex = 1;
                this.picCover.TabStop = false;
                //
                // tabDecoding
                //
                this.tabDecoding.BackColor = System.Drawing.Color.White;
                this.tabDecoding.Controls.Add(this.linkLabel1);
                this.tabDecoding.Controls.Add(this.button1);
                this.tabDecoding.Controls.Add(this.pictureBox2);
                this.tabDecoding.Controls.Add(this.grpMessageDec);
                this.tabDecoding.Controls.Add(this.btnDecode);
                this.tabDecoding.Controls.Add(this.txtPasswordDec);
                this.tabDecoding.Controls.Add(this.lblPasswordDec);
                this.tabDecoding.Controls.Add(this.btnStegoFile);
                this.tabDecoding.Controls.Add(this.txtStegoFileDec);
                this.tabDecoding.Controls.Add(this.lblStegoFile);
                this.tabDecoding.Controls.Add(this.groupBox3);
                this.tabDecoding.Font = new System.Drawing.Font("Comic Sans MS",
        9.75F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
        ((byte)(0)));
                this.tabDecoding.ForeColor = System.Drawing.Color.Indigo;
```

```
            this.tabDecoding.Location = new System.Drawing.Point(4, 27);
            this.tabDecoding.Name = "tabDecoding";
            this.tabDecoding.Size = new System.Drawing.Size(607, 507);
            this.tabDecoding.TabIndex = 1;
            this.tabDecoding.Text = "Decoding";
            //
            // linkLabel1
            //
            this.linkLabel1.Anchor =
    ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Bottom |
    System.Windows.Forms.AnchorStyles.Right)));
            this.linkLabel1.AutoSize = true;
            this.linkLabel1.BackColor = System.Drawing.Color.White;
            this.linkLabel1.ForeColor = System.Drawing.Color.Indigo;
            this.linkLabel1.Location = new System.Drawing.Point(3, 3);
            this.linkLabel1.Name = "linkLabel1";
            this.linkLabel1.Size = new System.Drawing.Size(165, 18);
            this.linkLabel1.TabIndex = 33;
            this.linkLabel1.TabStop = true;
            this.linkLabel1.Text = "Copyright © Pawan Gosavi";
            this.linkLabel1.TextAlign =
    System.Drawing.ContentAlignment.BottomRight;
            this.linkLabel1.LinkClicked += new
    System.Windows.Forms.LinkLabelLinkClickedEventHandler(this.linkLabel1_LinkClicked
    );
            //
            // button1
            //
            this.button1.BackColor = System.Drawing.Color.White;
            this.button1.Font = new System.Drawing.Font("Comic Sans MS", 8.25F,
    System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.button1.ForeColor = System.Drawing.Color.Indigo;
            this.button1.Location = new System.Drawing.Point(529, 3);
            this.button1.Name = "button1";
            this.button1.Size = new System.Drawing.Size(75, 26);
            this.button1.TabIndex = 32;
            this.button1.Text = "Exit";
            this.button1.UseVisualStyleBackColor = false;
            this.button1.Click += new System.EventHandler(this.button1_Click);
            //
            // pictureBox2
            //
            this.pictureBox2.Image =
    ((System.Drawing.Image)(resources.GetObject("pictureBox2.Image")));
            this.pictureBox2.Location = new System.Drawing.Point(192, 83);
            this.pictureBox2.Name = "pictureBox2";
            this.pictureBox2.Size = new System.Drawing.Size(217, 110);
            this.pictureBox2.SizeMode =
    System.Windows.Forms.PictureBoxSizeMode.StretchImage;
            this.pictureBox2.TabIndex = 27;
            this.pictureBox2.TabStop = false;
            //
            // grpMessageDec
            //
            this.grpMessageDec.Controls.Add(this.txtMessageDec);
            this.grpMessageDec.Font = new System.Drawing.Font("Comic Sans MS",
    9.75F, ((System.Drawing.FontStyle)((System.Drawing.FontStyle.Bold |
```

```
System.Drawing.FontStyle.Italic))), System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.grpMessageDec.ForeColor = System.Drawing.Color.Indigo;
            this.grpMessageDec.Location = new System.Drawing.Point(30, 325);
            this.grpMessageDec.Name = "grpMessageDec";
            this.grpMessageDec.Size = new System.Drawing.Size(550, 136);
            this.grpMessageDec.TabIndex = 18;
            this.grpMessageDec.TabStop = false;
            this.grpMessageDec.Text = " Hidden message";
            //
            // txtMessageDec
            //
            this.txtMessageDec.Font = new System.Drawing.Font("Comic Sans MS",
8.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.txtMessageDec.ForeColor = System.Drawing.Color.Red;
            this.txtMessageDec.Location = new System.Drawing.Point(6, 25);
            this.txtMessageDec.Multiline = true;
            this.txtMessageDec.Name = "txtMessageDec";
            this.txtMessageDec.ScrollBars =
System.Windows.Forms.ScrollBars.Vertical;
            this.txtMessageDec.Size = new System.Drawing.Size(538, 105);
            this.txtMessageDec.TabIndex = 4;
            //
            // btnDecode
            //
            this.btnDecode.BackColor = System.Drawing.Color.White;
            this.btnDecode.Font = new System.Drawing.Font("Comic Sans MS", 9.75F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.btnDecode.ForeColor = System.Drawing.Color.Indigo;
            this.btnDecode.Location = new System.Drawing.Point(508, 277);
            this.btnDecode.Name = "btnDecode";
            this.btnDecode.Size = new System.Drawing.Size(75, 30);
            this.btnDecode.TabIndex = 17;
            this.btnDecode.Text = "Decrypt";
            this.btnDecode.UseVisualStyleBackColor = false;
            this.btnDecode.Click += new
System.EventHandler(this.btnDecode_Click);
            //
            // txtPasswordDec
            //
            this.txtPasswordDec.ForeColor = System.Drawing.Color.Red;
            this.txtPasswordDec.Location = new System.Drawing.Point(164, 277);
            this.txtPasswordDec.Name = "txtPasswordDec";
            this.txtPasswordDec.PasswordChar = '*';
            this.txtPasswordDec.Size = new System.Drawing.Size(328, 26);
            this.txtPasswordDec.TabIndex = 16;
            //
            // lblPasswordDec
            //
            this.lblPasswordDec.Font = new System.Drawing.Font("Comic Sans MS",
9.75F, ((System.Drawing.FontStyle)((System.Drawing.FontStyle.Bold |
System.Drawing.FontStyle.Italic))), System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.lblPasswordDec.ForeColor = System.Drawing.Color.Indigo;
            this.lblPasswordDec.Location = new System.Drawing.Point(14, 277);
            this.lblPasswordDec.Name = "lblPasswordDec";
            this.lblPasswordDec.Size = new System.Drawing.Size(144, 30);
```

```
            this.lblPasswordDec.TabIndex = 15;
            this.lblPasswordDec.Text = " Provide Password :";
            this.lblPasswordDec.TextAlign =
    System.Drawing.ContentAlignment.MiddleCenter;
            //
            // btnStegoFile
            //
            this.btnStegoFile.BackColor = System.Drawing.Color.White;
            this.btnStegoFile.Font = new System.Drawing.Font("Comic Sans MS",
    9.75F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
    ((byte)(0)));
            this.btnStegoFile.ForeColor = System.Drawing.Color.Indigo;
            this.btnStegoFile.Location = new System.Drawing.Point(508, 229);
            this.btnStegoFile.Name = "btnStegoFile";
            this.btnStegoFile.Size = new System.Drawing.Size(75, 30);
            this.btnStegoFile.TabIndex = 13;
            this.btnStegoFile.Text = "Browse";
            this.btnStegoFile.UseVisualStyleBackColor = false;
            this.btnStegoFile.Click += new
    System.EventHandler(this.btnStegoFile_Click);
            //
            // txtStegoFileDec
            //
            this.txtStegoFileDec.ForeColor = System.Drawing.Color.Red;
            this.txtStegoFileDec.Location = new System.Drawing.Point(164, 229);
            this.txtStegoFileDec.Name = "txtStegoFileDec";
            this.txtStegoFileDec.Size = new System.Drawing.Size(328, 26);
            this.txtStegoFileDec.TabIndex = 12;
            //
            // lblStegoFile
            //
            this.lblStegoFile.Font = new System.Drawing.Font("Comic Sans MS",
    9.75F, ((System.Drawing.FontStyle)((System.Drawing.FontStyle.Bold |
    System.Drawing.FontStyle.Italic))), System.Drawing.GraphicsUnit.Point,
    ((byte)(0)));
            this.lblStegoFile.ForeColor = System.Drawing.Color.Indigo;
            this.lblStegoFile.Location = new System.Drawing.Point(14, 229);
            this.lblStegoFile.Name = "lblStegoFile";
            this.lblStegoFile.Size = new System.Drawing.Size(138, 30);
            this.lblStegoFile.TabIndex = 11;
            this.lblStegoFile.Text = " Select Stego File :";
            this.lblStegoFile.TextAlign =
    System.Drawing.ContentAlignment.MiddleCenter;
            //
            // groupBox3
            //
            this.groupBox3.Controls.Add(this.picStegoFileDec);
            this.groupBox3.Font = new System.Drawing.Font("Comic Sans MS", 9.75F,
    ((System.Drawing.FontStyle)((System.Drawing.FontStyle.Bold |
    System.Drawing.FontStyle.Italic))), System.Drawing.GraphicsUnit.Point,
    ((byte)(0)));
            this.groupBox3.ForeColor = System.Drawing.Color.Indigo;
            this.groupBox3.Location = new System.Drawing.Point(18, 42);
            this.groupBox3.Name = "groupBox3";
            this.groupBox3.Size = new System.Drawing.Size(166, 184);
            this.groupBox3.TabIndex = 10;
            this.groupBox3.TabStop = false;
            this.groupBox3.Text = "Stego File";
```

```
            //
            // picStegoFileDec
            //
            this.picStegoFileDec.BackColor = System.Drawing.Color.White;
            this.picStegoFileDec.Location = new System.Drawing.Point(6, 25);
            this.picStegoFileDec.Name = "picStegoFileDec";
            this.picStegoFileDec.Size = new System.Drawing.Size(150, 150);
            this.picStegoFileDec.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.StretchImage;
            this.picStegoFileDec.TabIndex = 1;
            this.picStegoFileDec.TabStop = false;
            //
            // Form1
            //
            this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);
            this.BackColor = System.Drawing.Color.White;
            this.ClientSize = new System.Drawing.Size(618, 541);
            this.Controls.Add(this.tabControl1);
            this.Icon =
((System.Drawing.Icon)(resources.GetObject("$this.Icon")));
            this.Name = "Form1";
            this.Text = "SteganoTech - Text Steganography
- By Pawan Gosavi" +
    "";
            this.tabControl1.ResumeLayout(false);
            this.tabEncoding.ResumeLayout(false);
            this.tabEncoding.PerformLayout();

((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).EndInit();
            this.grpMessageEnc.ResumeLayout(false);
            this.grpMessageEnc.PerformLayout();
            this.groupBox2.ResumeLayout(false);

((System.ComponentModel.ISupportInitialize)(this.picStegoFileEnc)).EndInit();
            this.groupBox1.ResumeLayout(false);

((System.ComponentModel.ISupportInitialize)(this.picCover)).EndInit();
            this.tabDecoding.ResumeLayout(false);
            this.tabDecoding.PerformLayout();

((System.ComponentModel.ISupportInitialize)(this.pictureBox2)).EndInit();
            this.grpMessageDec.ResumeLayout(false);
            this.grpMessageDec.PerformLayout();
            this.groupBox3.ResumeLayout(false);

((System.ComponentModel.ISupportInitialize)(this.picStegoFileDec)).EndInit();
            this.ResumeLayout(false);

            }
            #endregion

            /// <summary>
            /// The main entry point for the application.
            /// </summary>
            [STAThread]
            static void Main()
            {
                  Application.Run(new Form1());
```

```
}


private void btnCover_Click(object sender, System.EventArgs e)
{
      OpenFileDialog ofd = new OpenFileDialog();
      ofd.Multiselect = false;
      ofd.Filter = "Windows Bitmap (*.bmp)|*.bmp";
      if( ofd.ShowDialog(this) != DialogResult.Cancel)
      {
            txtCover.Text = ofd.FileName;
            Image pic = new Bitmap(txtCover.Text);
            FitPic(pic, picCover);
            picCover.Image = new Bitmap(pic);
            pic.Dispose();

      }
}

private void FitPic(Image pic, PictureBox box)
{
      box.Width = 150;
      box.Height = 150;

      int width = pic.Width;
      int height = pic.Height;

      if(width > height)
      {
            int size = box.Width;
            int x = height * size / width;
            box.Height = x;
      }
      else
      {
            int size = box.Height;
            int x = width * size / height;
            box.Width = x;
      }
}

private void btnEncode_Click(object sender, System.EventArgs e)
{
      string pic = txtCover.Text;
      string message = txtMessageEnc.Text;
      string password = txtPasswordEnc.Text;
      string stegoFile = txtStegoFileEnc.Text;

      if(pic == "")
      {
            MessageBox.Show("Select the cover");
            return;
      }

      if(message == "")
      {
            MessageBox.Show("Type the message to hide");
            return;
```

```
            }

            if(stegoFile == "")
            {
                  MessageBox.Show("Type the name of the stego file");
                  return;
            }

            if(!stegoFile.ToLower().EndsWith(".bmp"))
                  stegoFile += ".bmp";

            try
            {

                  // Open the cover
                  ICoverFile cover = new BMPCoverFile(pic);

                  // Create the stego file
                  cover.CreateStegoFile(stegoFile, message, password);

                  Result("Message hidden successfully");

                  Image stegoPic = new Bitmap(stegoFile);
                  FitPic(stegoPic, picStegoFileEnc);
                  picStegoFileEnc.Image = new Bitmap(stegoPic);
                  stegoPic.Dispose();

            } catch(Exception ex) {

                  Result(ex.Message);

            }

      }

      private void Result(string message)
      {

            MessageBox.Show(message);

      }

      private void btnDecode_Click(object sender, System.EventArgs e)
      {
            string stegoFile = txtStegoFileDec.Text;
            string password = txtPasswordDec.Text;

            if(stegoFile == "")
            {
                  MessageBox.Show("Select the stego file to decode");
                  return;
            }


            try {

                  // Open the stego file
                  IStegoFile stego = new BMPStegoFile(stegoFile, password);
```

```
                        // Show the hidden message
                        txtMessageDec.Text = stego.HiddenMessage;

                    }  catch(Exception ex) {

                        Result(ex.Message);

                    }

                }

                private void btnStegoFile_Click(object sender, System.EventArgs e)
                {
                    OpenFileDialog ofd = new OpenFileDialog();
                    ofd.Multiselect = false;
                    ofd.Filter = "Windows Bitmap (*.bmp)|*.bmp";
                    if( ofd.ShowDialog(this) != DialogResult.Cancel)
                    {
                        txtStegoFileDec.Text = ofd.FileName;
                        Image pic = new Bitmap(txtStegoFileDec.Text);
                        FitPic(pic, picStegoFileDec);
                        picStegoFileDec.Image = new Bitmap(pic);
                        pic.Dispose();

                    }
                }

                private void btnStegoFileEnc_Click(object sender, System.EventArgs e)
                {
                    OpenFileDialog ofd = new OpenFileDialog();

                    ofd.Filter = "Windows Bitmap (*.bmp)|*.bmp";
                    ofd.RestoreDirectory = true ;
                    ofd.CheckFileExists = false;

                    if(ofd.ShowDialog() == DialogResult.OK)
                    {
                        txtStegoFileEnc.Text = ofd.FileName;
                    }

                }

                private void tabControl1_SelectedIndexChanged(object sender,
        System.EventArgs e)
                {

                }

            private void button4_Click(object sender, EventArgs e)
            {

            }

            private void button3_Click(object sender, EventArgs e)
            {
                this.Close();
            }
```

```
        private void button2_Click(object sender, EventArgs e)
        {

        }

        private void button1_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void linkLabel2_LinkClicked(object sender,
LinkLabelLinkClickedEventArgs e)
        {

System.Diagnostics.Process.Start("https://plus.google.com/u/0/104671997220884711082");
        }

        private void linkLabel1_LinkClicked(object sender,
LinkLabelLinkClickedEventArgs e)
        {

System.Diagnostics.Process.Start("https://plus.google.com/u/0/104671997220884711082");
        }
    }
}
```

# User Manual

This is the first screen which is Login Page.

Provide The Only Username & Password Provided to You, In this case,

Username : Admin
Password : admin

Press Login, for Clearing Fields, use Clear, to Close window, use Exit.


After that, This is the Second screen you will see.

This is an Option Windows, It gives You two Option, Choose any one.

Picture Steganography : Hide Any File into a Image using Bitmap Format
Text Steganography : Hide Any Text into a Image using Password Encryption

After That You will See Either of these Below Windows, According to What you Choose.

## A] Image Steganography

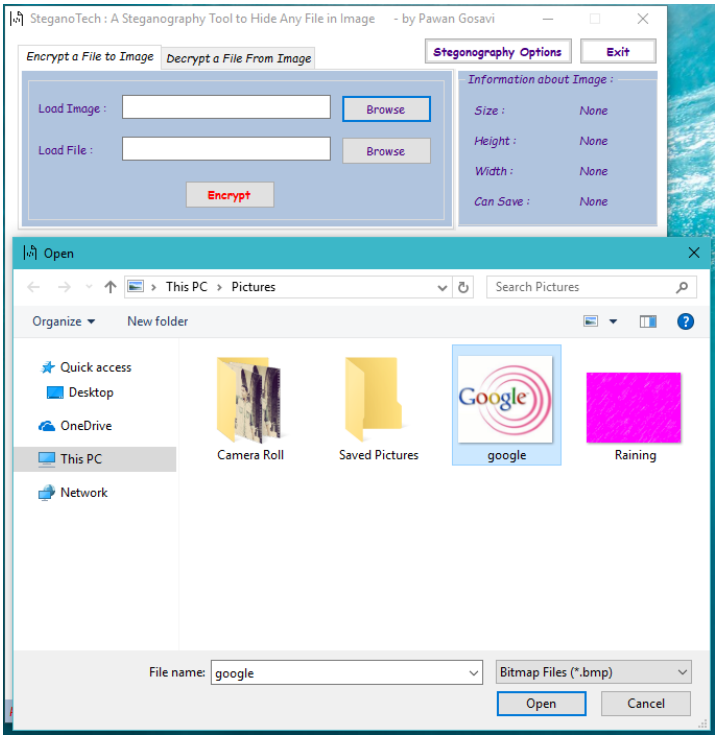
– one is Encrypt Image for encryption and another is Decrypt image for decryption.

In right – top panel is displays the information about the image such as size, height and width.

## Encryption

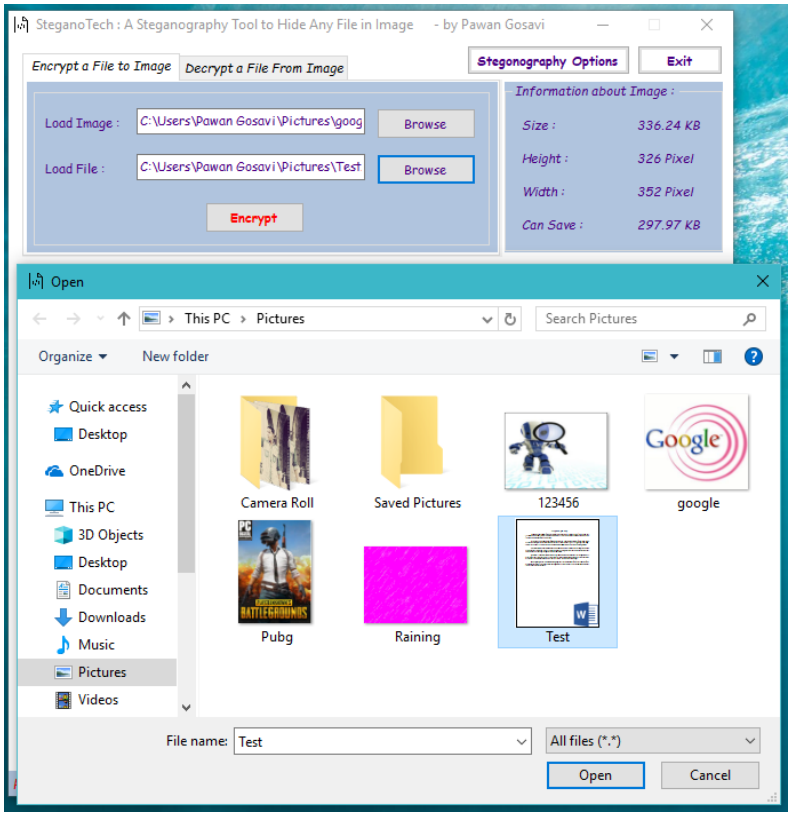
1. For Encryption select Encrypt Image tab option.

2. For load image click on button “Browse” that is next to the Load Image textbox. The file open dialog box will displays as follows, select the Image file, which you want to use hide information and click on Open button.

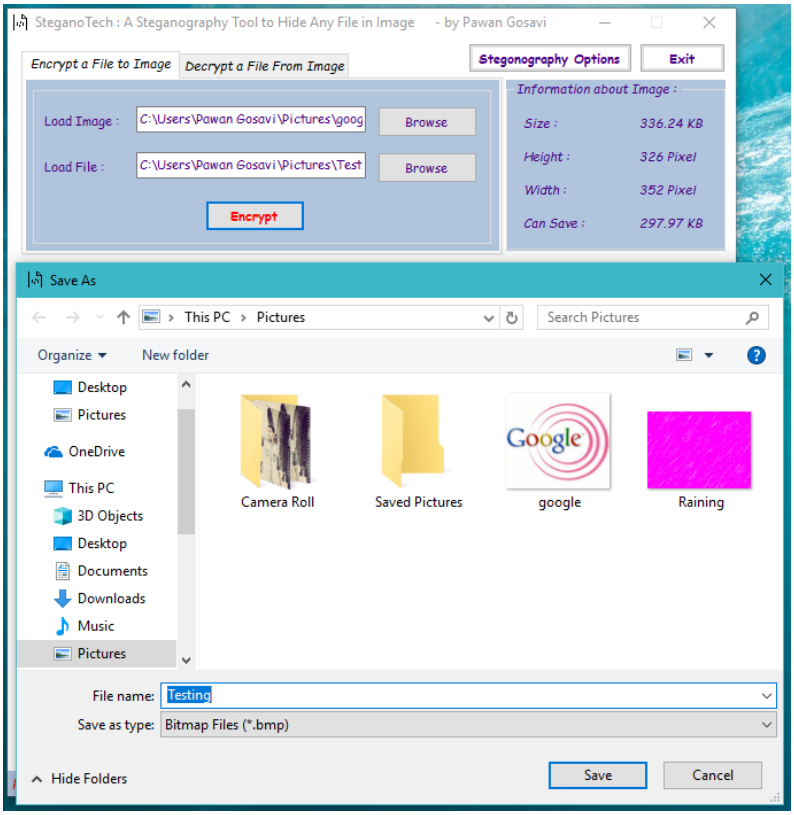



3. The image file will opened and is displays as follows. Next, click on “Browse” button that is next to the Load File textbox.

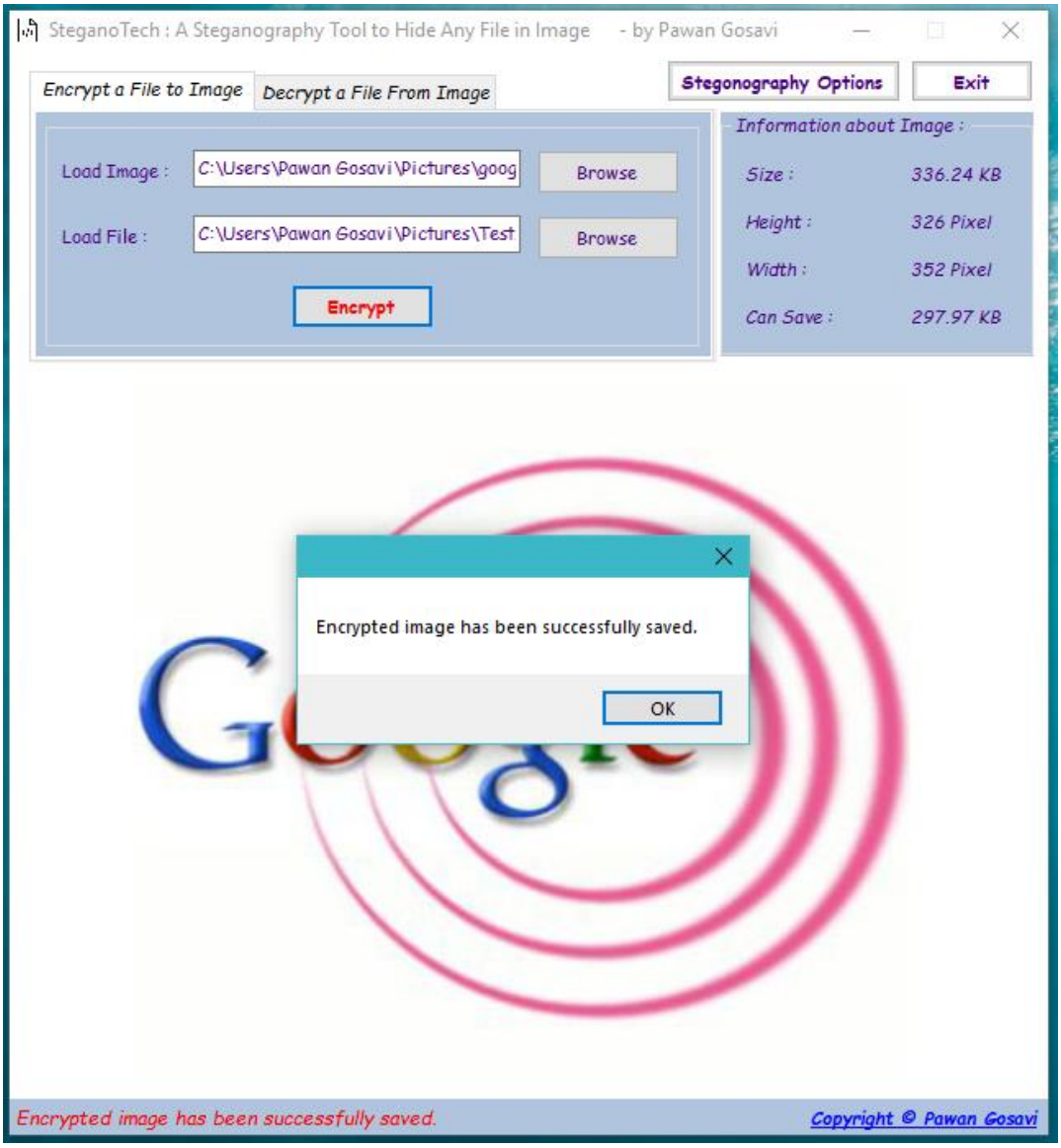
4. **Again the file open dialog box will appear, select any type of file whatever you want to hide with the image and click on ok button.**




5. **The next step is to encrypt the file. Now click on "Encrypt" button, it will open the save dialog box which ask you to select the path to save the New image file and the Image file name. The default format of image file is BMP.**



**Result:**




**Original Image:**

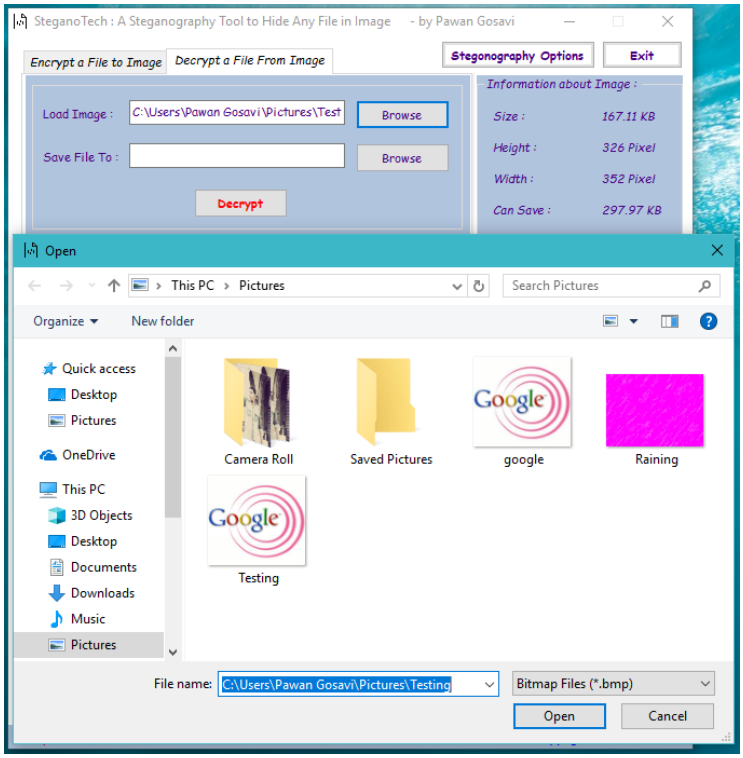

**Opening Encrypted Image :**


**See? No Difference at All.**

# Decryption

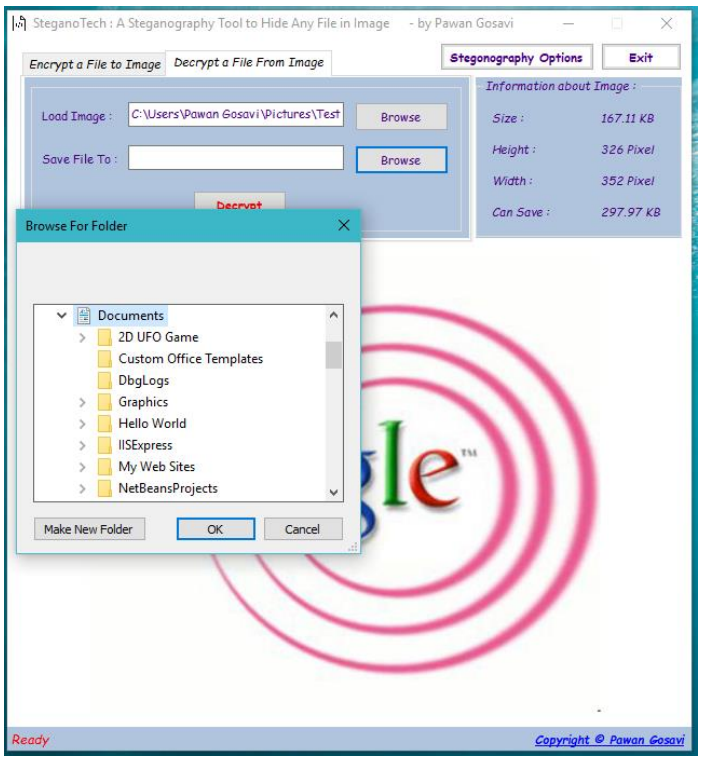
1. Select the Decryption Image tab option.

2. Next click on the “Browse” button, which open the Open file dialog box, here you have to select the image which is Encrypted and has hidden information file. Select the image file and click on Open button.



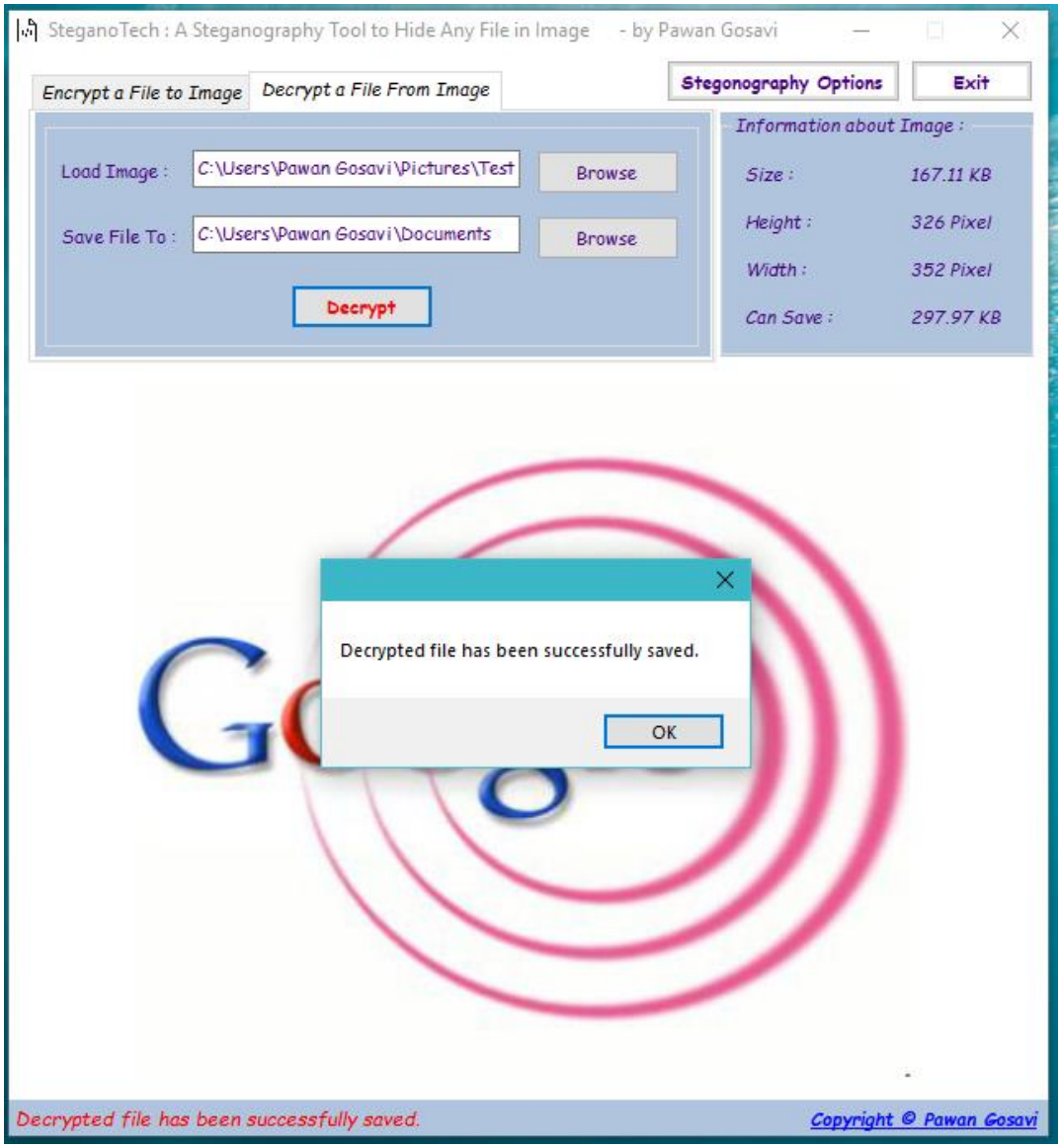

3. The image file displayed as follows:

4. Now click on “Browse” button which is next to “Save file to” textbox. It will open a dialog box that is “Browse for folder”. It ask you to select the path or folder, where you want to extract the hidden file. Select the folder and click on Ok button.

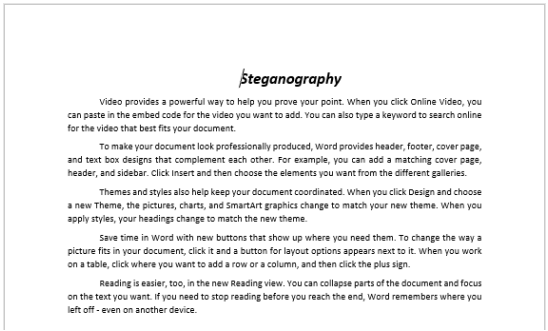



5. Now click on Decrypt button, it will decrypt the image, the hidden file and image file is saved into selected folder. The message for successful decryption is displayed on the status bar which is places at bottom of the screen.

**Result:**




**Original File:**



*steganography*

Video provides a powerful way to help you prove your point. When you click Online Video, you can paste in the embed code for the video you want to add. You can also type a keyword to search online for the video that best fits your document.

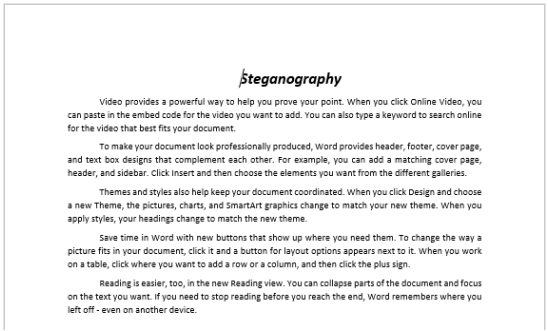
To make your document look professionally produced, Word provides header, footer, cover page, and text box designs that complement each other. For example, you can add a matching cover page, header, and sidebar. Click Insert and then choose the elements you want from the different galleries.

Themes and styles also help keep your document coordinated. When you click Design and choose a new Theme, the pictures, charts, and SmartArt graphics change to match your new theme. When you apply styles, your headings change to match the new theme.

Save time in Word with new buttons that show up where you need them. To change the way a picture fits in your document, click it and a button for layout options appears next to it. When you work on a table, click where you want to add a row or a column, and then click the plus sign.

Reading is easier, too, in the new Reading view. You can collapse parts of the document and focus on the text you want. If you need to stop reading before you reach the end, Word remembers where you left off - even on another device.

**Opening Decrypted File :**



*steganography*

Video provides a powerful way to help you prove your point. When you click Online Video, you can paste in the embed code for the video you want to add. You can also type a keyword to search online for the video that best fits your document.

To make your document look professionally produced, Word provides header, footer, cover page, and text box designs that complement each other. For example, you can add a matching cover page, header, and sidebar. Click Insert and then choose the elements you want from the different galleries.

Themes and styles also help keep your document coordinated. When you click Design and choose a new Theme, the pictures, charts, and SmartArt graphics change to match your new theme. When you apply styles, your headings change to match the new theme.

Save time in Word with new buttons that show up where you need them. To change the way a picture fits in your document, click it and a button for layout options appears next to it. When you work on a table, click where you want to add a row or a column, and then click the plus sign.

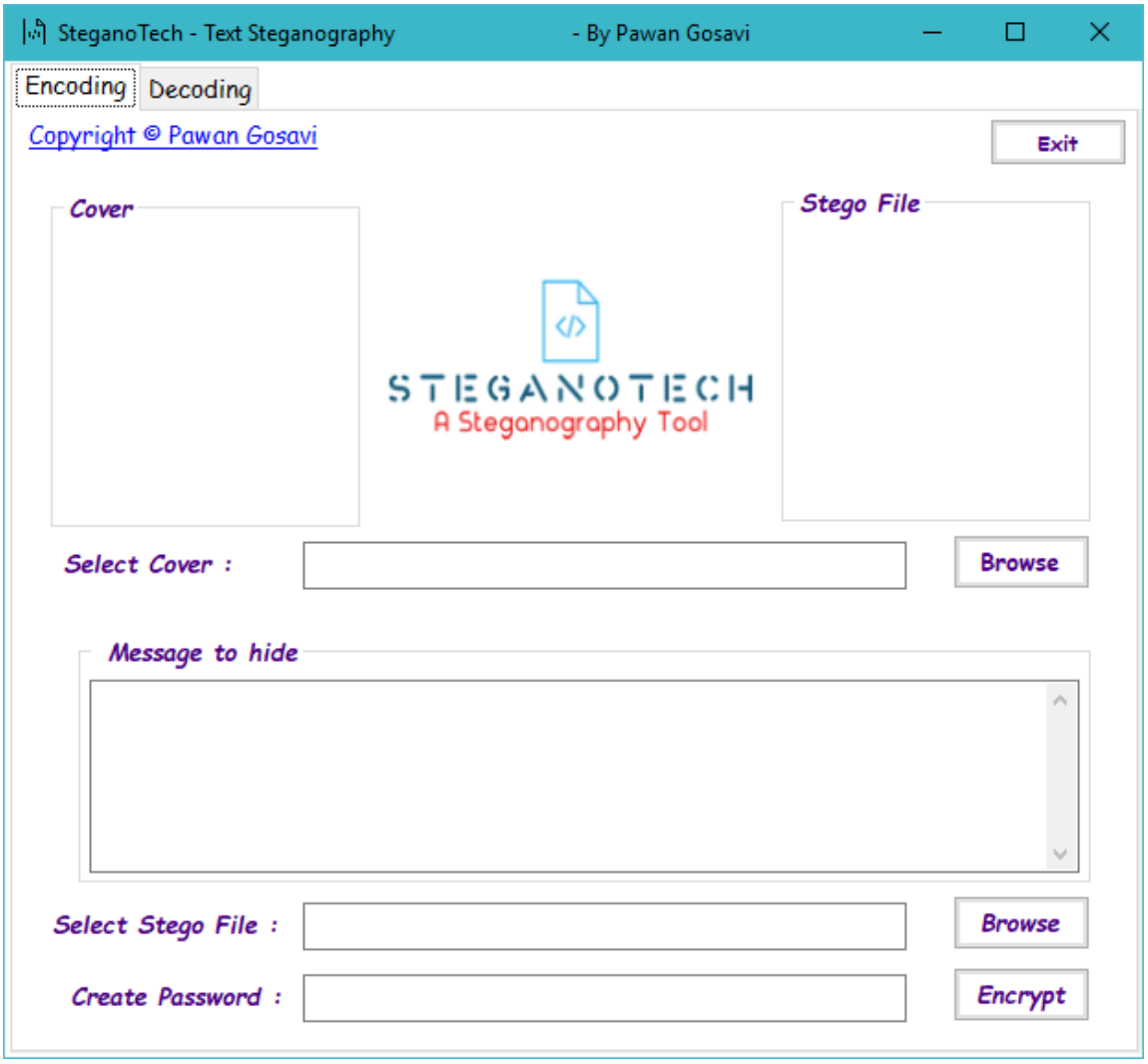
Reading is easier, too, in the new Reading view. You can collapse parts of the document and focus on the text you want. If you need to stop reading before you reach the end, Word remembers where you left off - even on another device.

**See? No Difference at All. Exactly Same File is recovered.**

## B] Text Steganography

– one is Encoding for encryption and another is Decoding for decryption.

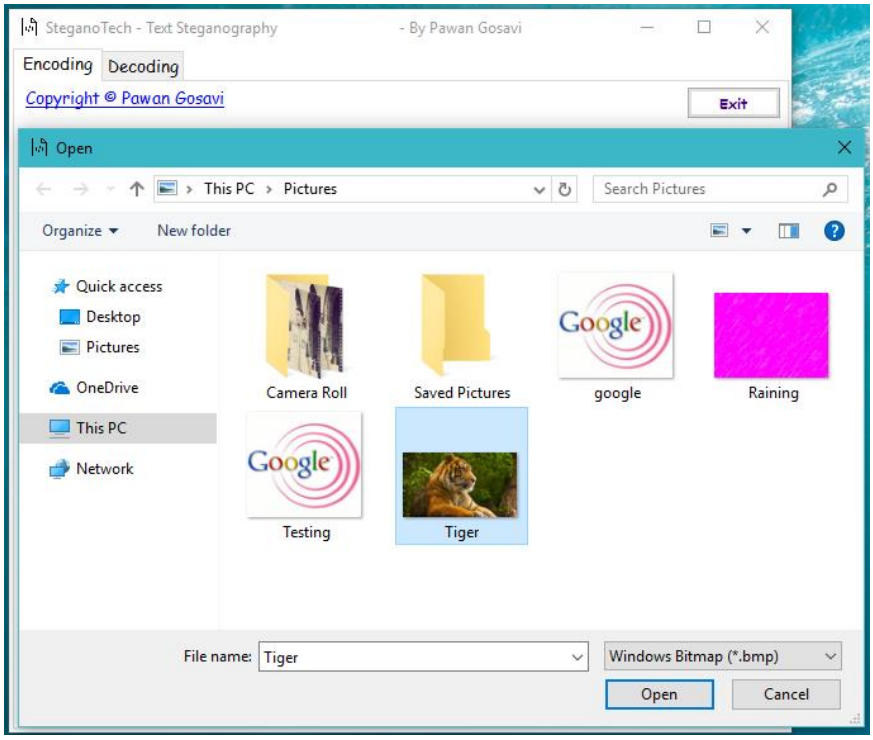
In right & left – top panel is displays the cover and the Stego File you will Choose.



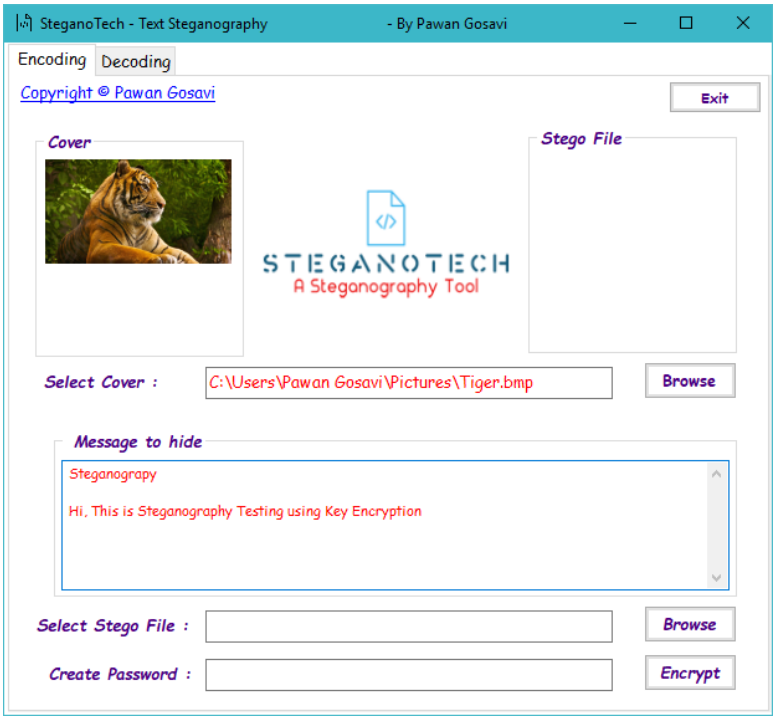
# Encoding

1. For Enccoding select Encoding tab option.

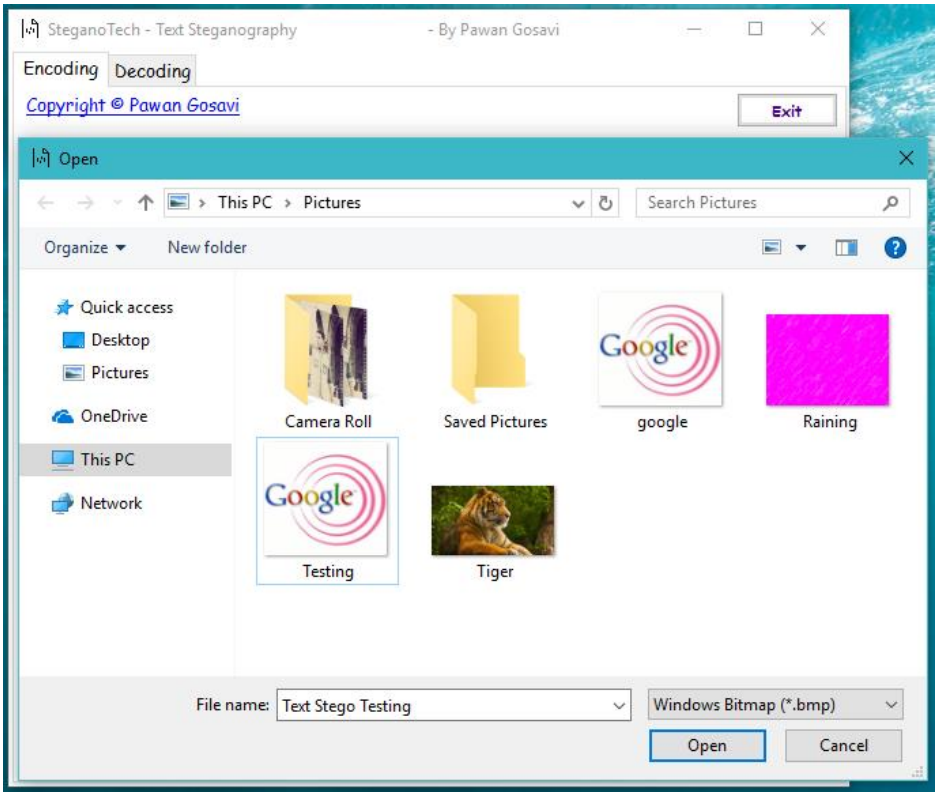
2. For Cover image click on button “Browse” that is next to the Select Cover textbox. The file open dialog box will displays as follows, select the Image file, which you want to use hide information and click on Open button.



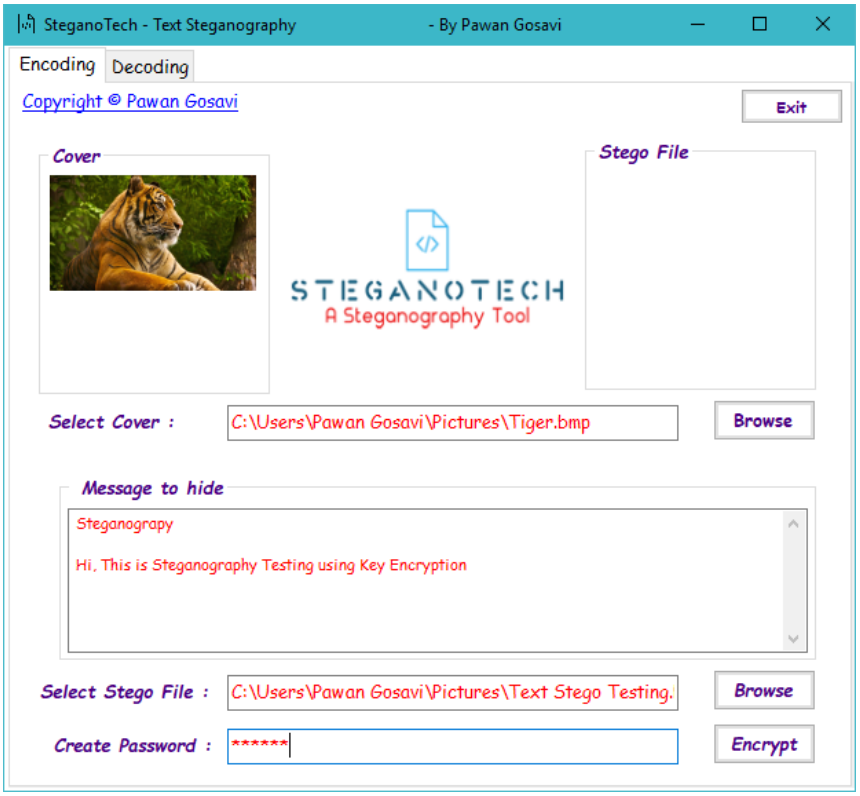

3. The image file will opened and is displays as follows. Next, Type the Message You want to hide in the Selected Image.



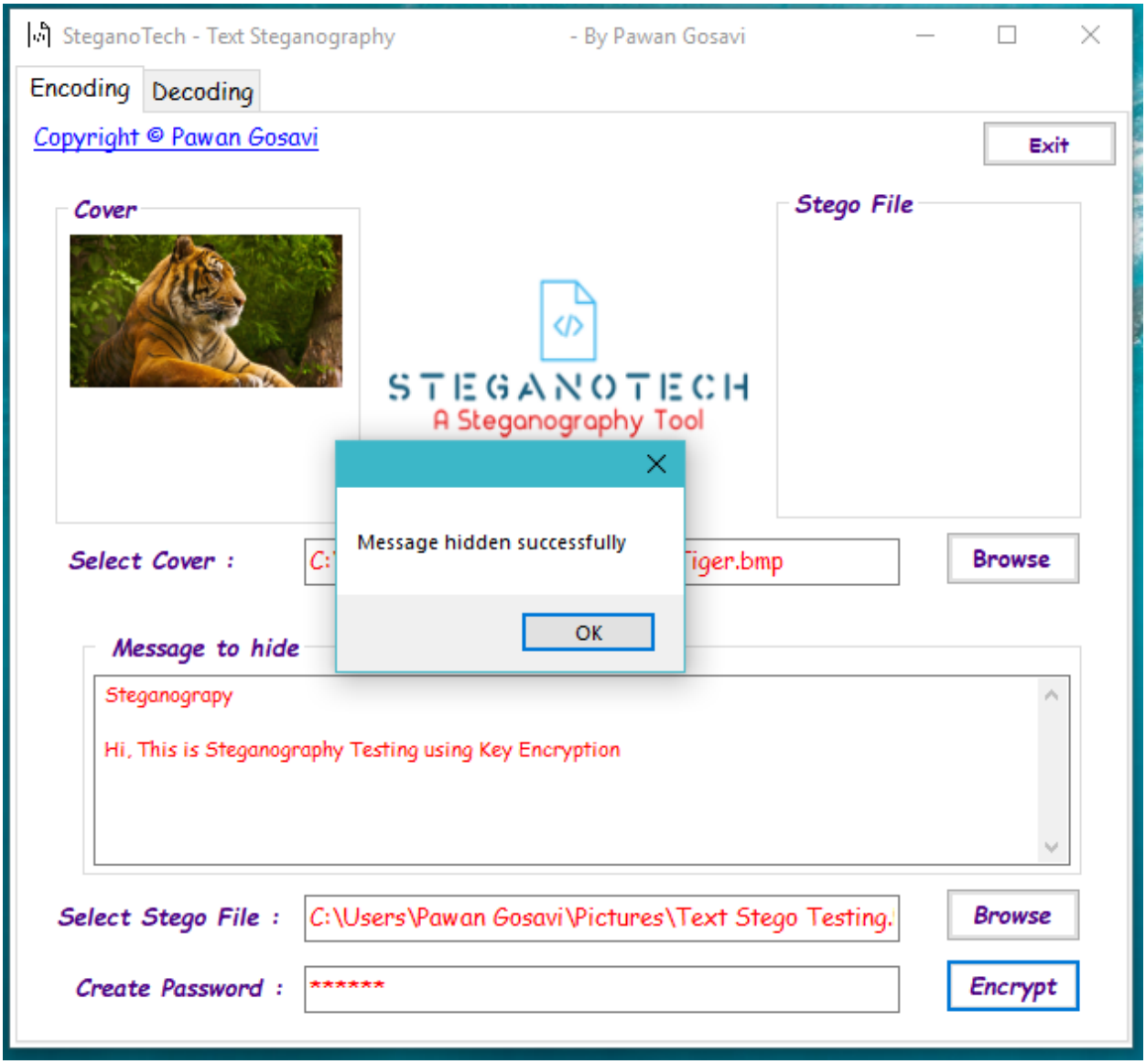
4. **Next, click on "Browse" button that is next to the Select Stego File textbox. Again the file open dialog box will appear, give any type of name whatever you want to Stego image and click on Open button.**




5. **The next step is to Create a Password to encrypt the file. Type Password in Password Field. Now click on "Encrypt" button, it will open Confirmation Box. The default format of image file is BMP.**



**Result:**




**Original Image:**

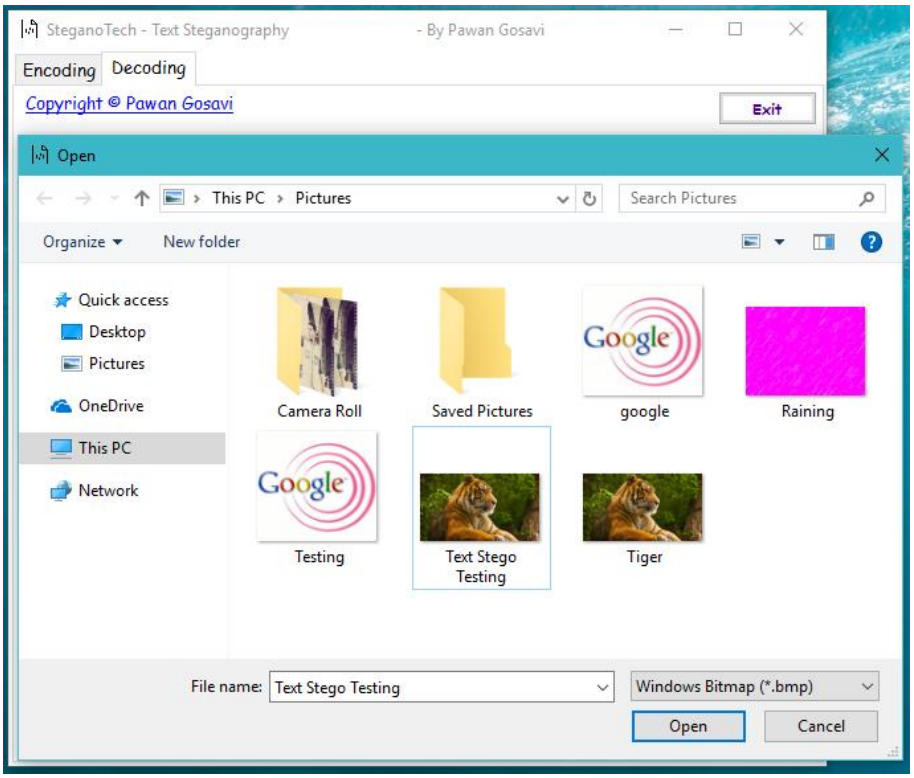
**Opening Encrypted Image :**

**See? No Difference at All.**

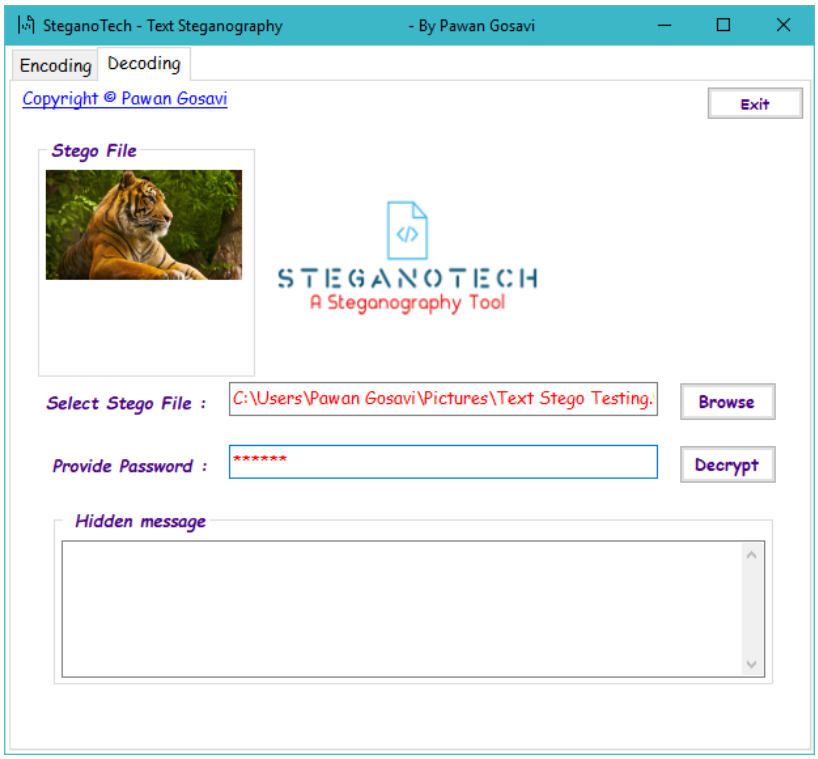
# Decryption

1. Select the Decoding Image tab option.

2. Next click on the “Browse” button, which open the Open file dialog box, here you have to select the image which is Encrypted and has hidden information file. Select the image file and click on Open button.

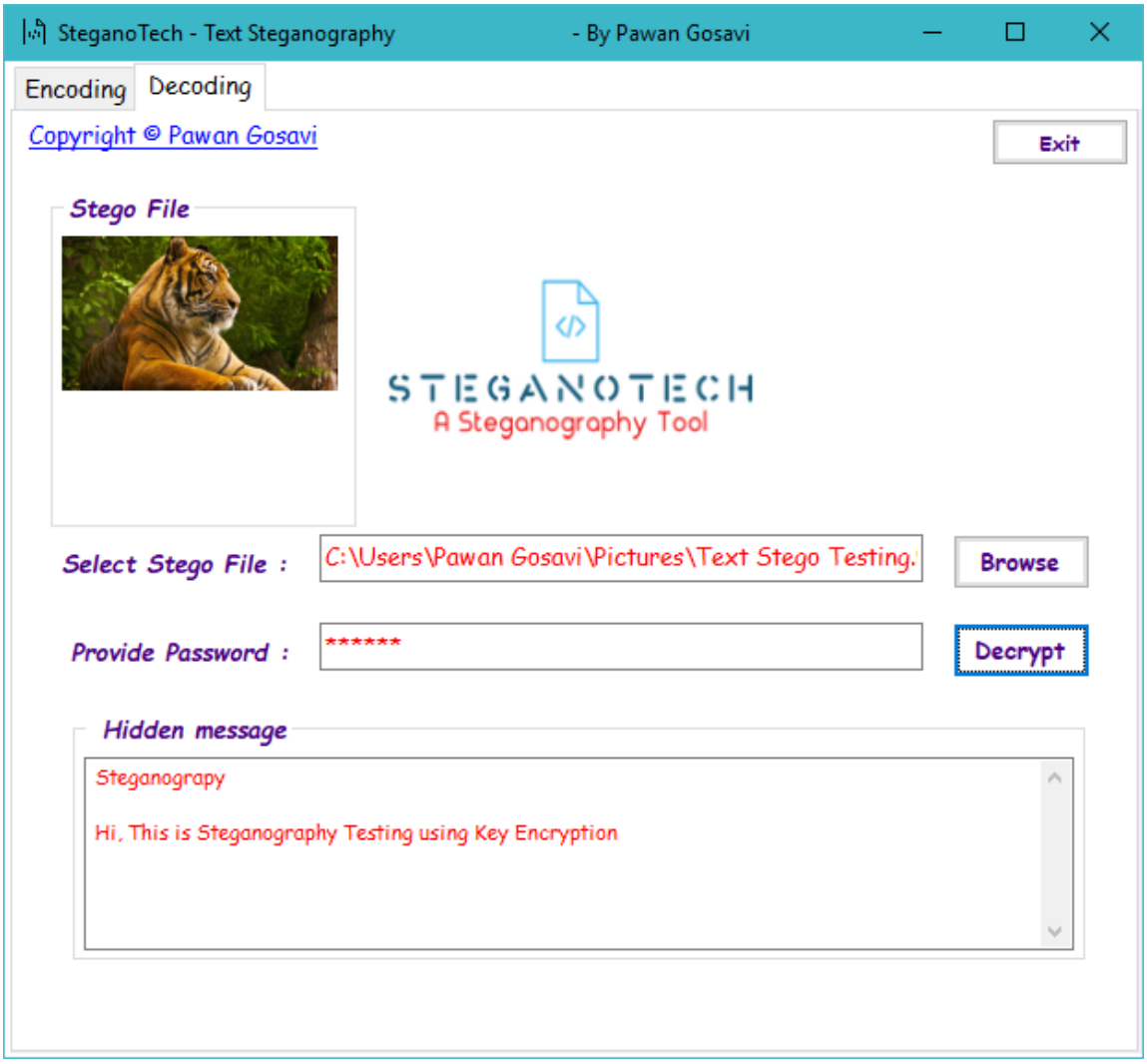



3. The image file displayed as follows. Enter The Password Provided to You.



4. **Now click on “Decrypt” button. If your Password is Correct You Can Now see The Text.**

**Result:**




**Original Text:**

Steganograpy

Hi, This is Steganography Testing using Key Encryption

**Decrypted Text :**

Steganograpy

Hi, This is Steganography Testing using Key Encryption

**See? No Difference at All. Exactly Same Text is recovered.**

# Testing

The test procedure used in the testing process is Black box testing. Test cases are analyzed accordingly.

## Black Box Testing

This test involves the manual evaluation of the flow from one module to the other and check accordingly for the process flow. This process of testing is with the following criteria

Browsing process
Encryption process
Retrieving Process
Decryption process
Authentication process
Key prescription Information display
Exit process

Case Generation Report:

| Test Case No. | Test Type | Case | Expected Result |
|---|---|---|---|
| 001 | Operational / Unit / Functional Test | Browse | Receive the file to load and Save. |
| 002 | - // - | Encryption | Receive the file to be encrypted and encrypt According to the Key and Save. |
| 003 | - // - | Decryption | Receive the file to be decrypted and decrypt with same Key and Save. |
| 004 | - // - | Exit | Ends the Process. |
| 005 | - // - | Retrieve | Receives the Folder contains Encrypted file to retrieve the Encrypted file and save in to the Another folder. |

Test Case Analysis:

| Test Case No. | Test Type | Expected Result | Observed Result | Remarks |
|---|---|---|---|---|
| 001 | Operational / Unit / Functional Test | Receive Decrypted data, Decrypt and save | Path failure | Address of the file is corrected. |

All the above validations on buttons have been verified and they are successfully executed. The flow is tested at different possible conditions by means of this testing.

## Limitations of the Software:

This project has an assumption that is both the sender and receiver must have shared some secret information before imprisonment. Pure steganography means that there is none prior information shared by two communication parties.

## Detecting Steganography:

The art of detecting Steganography is referred to as Steganolysis.

To put is simply Steganolysis involves detecting the use of Steganography inside of a file. Steganolysis does not deal with trying to decrypt the hidden information inside of a file, just discovering it.

There are many methods that can be used to detect Steganography such as:

"Viewing the file and comparing it to another copy of the file found on the Internet (Picture file)".

There are usually multiple copies of images on the internet, so you may want to look for several of them and try and compare the suspect file to them. For example if you download a JPED and your suspect file is also a JPED and the two files look almost identical apart from the fact that one is larger than the other, it is most probable you suspect file has hidden information inside of it.

## Future Enhancements:

To make it pure steganography application.

Also Adding Sound Steganography and Random Key Encryption Steganography

## Summary

Steganography is a really interesting subject and outside of the mainstream cryptography and system administration that most of us deal with day after day. Steganography can be used for hidden communication. We have explored the limits of steganography theory and practice. We printed out the enhancement of the image steganography system using LSB approach to provide a means of secure communication. A stego-key has been applied to the system during embedment of the message into the cover image.

This steganography application software provided for the purpose to how to use any type of image formats to hiding any type of files inside their. The master work of this application is in supporting any type of pictures without need to convert to bitmap, and lower limitation on file size to hide, because of using maximum memory space in pictures to hide the file.

Since ancient times, man has found a desire in the ability to communicate covertly. The recent explosion of research in watermarking to protect intellectual property is evidence that steganography is not just limited to military or espionage applications. Steganography, like cryptography, will play an increasing role in the future of secure communication in the “digital world”.

## Bibliography

### Websites

Following websites are referring to create this project reports.


http://www.google.com
http://www.microsoft.com
http://www.asp.net
http://www.asp123.com
http://www.wikipedia.org


### Books

Following books and ebook are used to complete this project reports.


Mastering C# (Paperback)
SQL Server Bible (Paperback)
.NET Black Book (Paperback)
Professional C#, 2nd Edition (Paperback)
Professional ASP.NET (Paperback)

## Plagarism Report

# Plagiarism Checker X Originality Report

**Similarity Found: 26%**

Date: Friday, October 19, 2018
Statistics: 3050 words Plagiarized / 11733 Total words
Remarks: Medium Plagiarism Detected - Your Document needs Critical Improvement.
---