

**PROJECT REPORT ON**  
**SECURE DATA TRANSFER THROUGH SOCKET**  
**PROGRAMMING USING IMAGE**  
**STEGANOGRAPHY TECHNIQUE -JAVA**

*Report submitted to the SASTRA Deemed to be University as the  
requirement for the course*

**CSE302: COMPUTER NETWORKS**

*Submitted by*

**CHENNAMSETTI GOPINATH**

**Reg. No: 124003069**

**B. Tech COMPUTER SCIENCE AND ENGINEERING**

**December 2022**



**SCHOOL OF COMPUTING**

**THANJAVUR, TAMILNADU, INDIA- 613 401**



## SCHOOL OF COMPUTING

THANJAVUR– 613401

### Bonafide Certificate

This is to certify that the report titled “**Secure data transfer through Socket Programming Using Image Steganography Technique- Java**” submitted as a requirement for the course, **CSE302: COMPUTER NETWORKS** for B.Tech. is a bonafide record of the work done by **Shri. CHENNAMSETTI GOPINATH (Reg.No.124003069, CSE)** during the academic year 2022-23, in the School of Computing.

Project Based Work *Viva voce* held on \_\_\_\_\_

**Examiner 1**

**Examiner 2**

## LIST OF FIGURES

Figure No.	Title	Page No.
1	Basic Steganography Model	1
2	Types of Steganography	4
3	Steganography Logic	4
4	Protocol Function utilized by Steganography	5
5	Datalink Layer	6
6	Application Layer	6
7	Subnet Masking	7
8	Transport Layer	8
9	State Diagram of Client and Server	10
10	Proposed System	14
11	Class and State Diagram	15
12	Use Case Diagram	15
13	Flow chart diagram for Encryption and Decryption	16
14	Flow chart diagram for login page	16
15	Implementation Snap shots	41
16	Output Snap shots	42-46

## LIST OF TABLES

Table no	Title	Page no
1	Socket class Methods	10
2	Server Socket class Methods	10

## ACKNOWLEDGEMENTS

First of all, I want to express my gratitude to the Almighty for his unending favours. I want to thank my parents and everyone else who encouraged me to become interested in the project and helped me stay motivated to finish it before the deadline with minimal effort.

I would like to express my sincere gratitude to **Dr S. Vaidya Subramaniam**, Vice-Chancellor for his encouragement during the span of my academic life at SASTRA Deemed University.

I would forever remain grateful and I would like to thank **Dr A. Uma Makeswri**, Dean, School of Computing and **R. Chandramouli**, Registrar for their overwhelming support provided during my course span in SASTRA Deemed University.

I would specially thank and express my gratitude to **Dr. Shankar Sriram** , Associate Dean, School of Computing for providing me an opportunity to do this project and for his guidance and support to successfully complete the project and also academic help extended for the past two years of my life in School of computing

I would specially thank and express my gratitude to **Prof. Sujeet Jagtap S**, School of Computing for his guidance and support to successfully complete the project.

I also thank all the Teaching and Non-teaching faculty, and all other people who have directly or indirectly help me through their support, encouragement and all other assistance extended for completion of my project and for successful completion of all courses during my academic life at SASTRA Deemed University.

Finally, I thank my parents and all others who help me acquire this interest in project and aided me in completing it within the deadline without much struggle.

## **ABBREVIATIONS**

OSI -Open Systems Interconnections

LAN -Local Area Networks

WAN -Wide Area Networks

MSB -Most Significant Bits

LSB -Least Significant Bits

TCP -Transmission Control Protocols

IP -Internet Protocols

MAC- Media Access Control

ARP -Address Resolution Protocol

TCP- Transmission Control Protocol

OSI -Open Systems InterConnection

## ABSTRACT

The Steganography is the study of concealing or embedding "data" in a transmission medium. One of its ultimate objectives is to become invisible. Data may now be sent to its destination more precisely and quickly because to the internet's rapid expansion of data transport. Data can be transferred through a variety of transmission mediums to places like e-mails and social networking sites. Meanwhile, hacking might make it simpler to change and abuse crucial data. As a result, Image Steganography is utilised to deliver data securely and unaltered to its destination. Writing that is concealed or covered is known as steganography. Steganography is a form of clandestine communication used to keep a message secret from a third party. This proposed system deals with implementing security of data using Steganography.

Using socket programming, client-server applications can be built and the Steganography Technique can be used. In this technique, the user selects the image that will serve as the data carrier. The data file is authenticated and encrypted. The picture conceals this message. The image will open in any image previewer if it has been hacked or intercepted by a third party user, but the data won't be shown. This prevents the data from becoming invisible and ensures that it is transmitted securely. The user on the receiving end uses additional code to extract the image's contents. For data concealment in steganography, we use image steganography. In addition, we utilise the Mutual Authentication procedure to ensure that all services in Cryptography i.e., Access Control, Confidentiality, Integrity, Authentication. In this way we can maintain the data more securely. Since we utilise the LSB technique to secure the data, we can execute Steganography on this data to conceal it in an image. so that no other network user can access the info that is already available. The message can only be extracted from the data by the sender and the recipient.

This project report discusses image steganography and goes into great detail about various steganographic algorithms, such as Least Significant Bit (LSB) The IDE is utilised, and the language is (JAVA) (Eclipse). The project also gives the client the option to log into the website and access network information including Mac addresses and subnet information. This information was calculated using Subnet Masking and MARP (Mac Address Resolution Protocol), which gave the hosts' MAC addresses with provided Ip address.

**Keywords :** Steganography, Encryption ,Decryption , Socket Programming, Address Resolution protocol , Subnet Masking

# TABLE OF CONTENTS

BONAFIDE CERTIFICATE	ii
LIST OF FIGURES	iii
LIST OF TABLES	iii
ACKNOWLEDGEMENT	iv
ABBREVIATIONS	v
ABSTRACT .....	vi
<b>CHAPTER ONE</b> .....	1
INTRODUCTION .....	1
1.1 BACKGROUND TO THE STUDY .....	1
1.2 PROBLEM STATEMENT .....	2
1.3 AIM AND OBJECTIVES.....	2
1.4 SIGNIFICANCE OF THE PROJECT .....	2
1.5 SCOPE AND LIMITATION OF THE STUDY.....	2
1.6 DEFINITION OF TERMS/ACRONYMS USED .....	3
1.7 ORGANIZATION OF REPORT .....	3
1.8 TYPES OF STEGANOGRAPHY.....	3
1.9 ILLUSTRATING LAYERS INVOLVED IN PROJECT.....	4
1.10 MERITS AND DEMERITS OF STEGANOGRAPHY.....	9
1.11 SOCKET PROGRAMMING IN JAVA.....	9.
<b>CHAPTER TWO</b> .....	11
LITERATURE REVIEW .....	11
OVERVIEW OF THE EXISTING RELATEDWORK.....	11

2.1 AN OVERVIEW OF THE ORIGIN OF STEGANOGRAPHY.....	11
2.2 AN OVERVIEW OF MODERN STEGANOGRAPHY	
<b>CHAPTER THREE .....</b>	<b>12</b>
METHODOLOGY.....	12
3.1 LSB ALGORITHM.....	12
3.1.1 SYSTEM ANALYSIS .....	13
3.1.2 SYSTEM DESIGN .....	13
3.2 ANALYSIS OF THE PROPOSED SYSTEM	12
3.2.1 PROPOSED ALGORITHM .....	14
3.3 CLASS AND STATE MODEL .....	15
3.3.1 USE CASE DIAGRAM.....	15
3.3.2 FLOW CHARTS OF THE PROJECT.....	16
<b>CHAPTER FOUR.....</b>	<b>17</b>
IMPLEMENTATION AND RESULTS.....	17
4.1.1 SOURCE CODE OF THE PROJECT.....	17
4.1.2 IMPLEMENTATION OF THE PROJECT....	41
4.1.3 RESULTS AND SNAPSHOTS.....	45
<b>CHAPTER FIVE.....</b>	<b>47</b>
5.1 SUMMARY .....	47
5.2 CONCLUSION .....	47
5.3 RECOMMENDATION.....	48
REFERENCES.....	48



# CHAPTER ONE

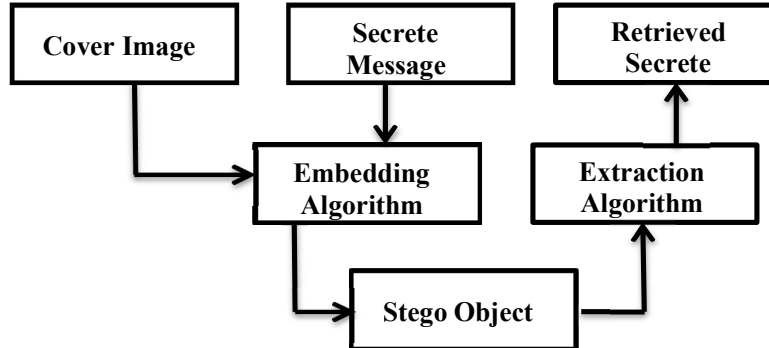
## INTRODUCTION

### 1.1 BACKGROUND TO THE STUDY

The security of information has been one of the most crucial aspects of communication and information technology since the advent of the Internet. This information must be safeguarded while being transmitted across insecure routes. Therefore, there is a need for creating technologies that would aid in safeguarding owners' intellectual property rights and protecting the integrity of digital material. As a result, the field of information concealment has grown dramatically. Additionally, the quick development of publishing and broadcasting technology necessitates a different approach to information concealment. To solve this issue, it may be possible to incorporate some invisible data in digital material in a way that makes it difficult to access it without specialist equipment. Therefore, this project study offered steganography, a data hiding technique, together with the encryption-decryption approach in order to create a superior security mechanism. Steganography is the most popular and commonly utilised method of data security.

**The word steganography is derived from the Greek words “stegos” meaning “cover” and “grafia” meaning “writing” defining it as “covered writing”.**

Steganography is primarily employed on computers today, with networks acting as the high-speed delivery channels and digital data serving as the carriers. It is a performance in which hidden messages are inserted into a cover file in order to conceal their presence.



**Fig. 1 Basic Steganography Model**

The cover object, message, embedding technique, and Stego key make up the fundamental components of steganography.

Cover item, often referred to as a carrier, serves to conceal the message and its presence. Secret data can be concealed using digital photos, movies, sound files, and other PC files that include perceptually pointless or redundant information. A "stego object" is created by embedding secret data within the cover-object. Applying an extraction at the receiver end Secret message on the cover image Secrete Message Extraction and Embedding Algorithm, Stego Object, retrieved. Using the Basic Steganography Model 3 technique, we are able to extract the hidden message from the Stego object (Fig. 1.1).

The image's visibility, resolution, or clarity are unaffected in this project job because the data is concealed utilising an image file. The concealed data may be large in scope. Only the image and no indication of the secret data will be revealed to the hacker when the image is previewed. Since the data is encoded in an encryption format, which is likewise binary, it will also be impossible for the enclosure to distinguish the data from the picture file if the image file is opened in a text editor.

## **1.2 PROBLEM STATEMENT**

It is crucial that we use a more secure approach (in this example, steganography), which conceals words in carriers like photos, because hiding messages using cryptography appears to be susceptible to cryptanalysis (albeit, relatively). Using the Least Significant Bit (LSB) technique, this work implements image steganography for a more secure data transport, particularly over the internet. The security issues that cryptography presents are thought to have a sure cure by using this method to develop image steganography..

## **1.3 AIM AND OBJECTIVES**

The aim of this project work is to develop a system that will implement steganography using image which serve as security measure by hidden communication to cover a message from third party. The objectives of the study are highlighted as below: i. To carry out a critical review of the concept of information security ii. To carry out a thorough survey of the various steganography types we have iii. To carry out an elaborate investigation into image steganography

## **1.4 SIGNIFICANCE OF THE PROJECT**

There is a need to come up with a way of communicating covertly that will help in overcoming this threat of insecurity met during data transfer due to the ongoing problem of security breach faced during information exchange (between a sender and a receiver). With steganography, it would take an invader an eternity to try to decipher the user-protected message. In other words, there is no question about the message's security..

## **1.5 SCOPE AND LIMITATION OF THE STUDY**

This Application uses Bitmap image (.bmp format), JPG (.jpg format), or GIF format as its carrier file to conceal data or information inside images (a method known as steganography). However, only texts can be hidden with this technique, and it might not work with other data formats. Passwords that must be shared can be used and exploited. The recipient must be manually notified of the image..

## 1.6 DEFINITION OF TERMS/ACRONYMS USED

**Audio/Video:** is similar to the two techniques above, only that it uses audio/video as a cover object to hide the existence of data.

**Images Steganography:** Due to its high bit redundancy, it is frequently used as a cover object to conceal data.

**The Least Significant Bits Protocol (LSB) :** uses network protocols and data-embedding techniques to conceal the existence of data during network transmission.

.

## 1.7 ORGANIZATION OF REPORT

This project work is divided into five (5) chapters and the descriptions of each are given below: -

**Chapter one:** This seeks to give an introduction to the above topic of consideration.

**Chapter two:** This chapter contains the literature review.

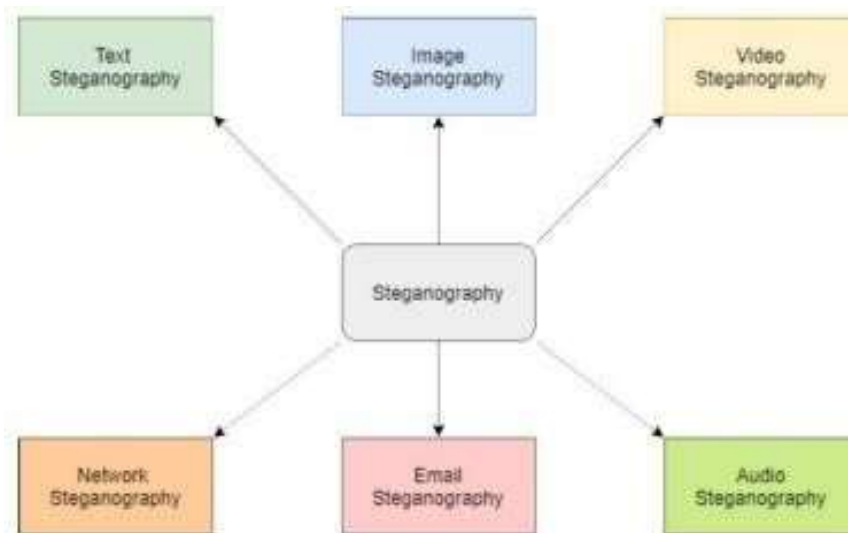
**Chapter three:** This is the general analysis of the system and the methodology employed.

**Chapter four:** This embodies the implementation and documentation of the newly designed system

**Chapter five:** This chapter contains the summary of the entire project, recommendation and conclusion

## 1.8 TYPES OF STEGANOGRAPHY

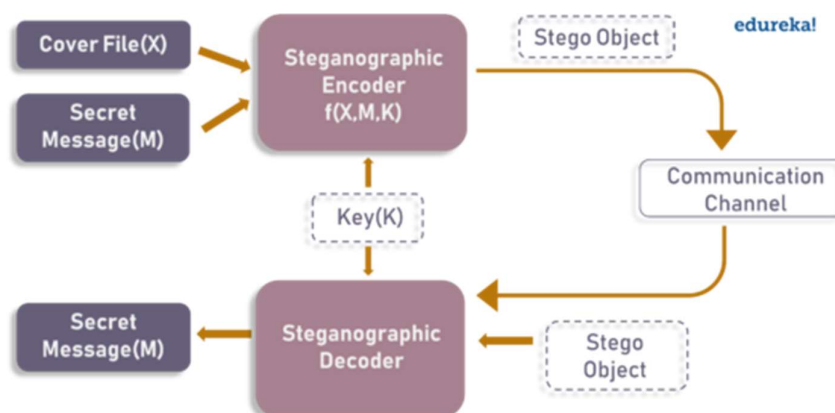
Based on the way of transmission steganography can be classified into **Text steganography, Image steganography, Video steganography, Network steganography, E-mail steganography, Audio steganography**



**Fig 2**Types of steganography

### Image Steganography

The most common cover item for steganography is an image. There are many steganographic algorithms for the various image file types. There are multiple steganography algorithms for these various image file types. The Image Domain and the Transform Domain are two categories into which image steganography techniques fall. While transform-domain techniques, also known as frequency-domain, first alter images before encoding a message, image-domain approaches directly embed messages in the intensity of the pixels. Image domain approaches are commonly referred to as "simple systems," and they include bit-wise methods that use bit insertion and noise manipulation. Lossless image formats and standard steganography methods are best suited for use in image domains. These methods hide messages in more significant areas of the cover image, making it more robust. Many transform domain methods are independent of the image format and the embedded message may survive conversion between lossy and lossless compression.



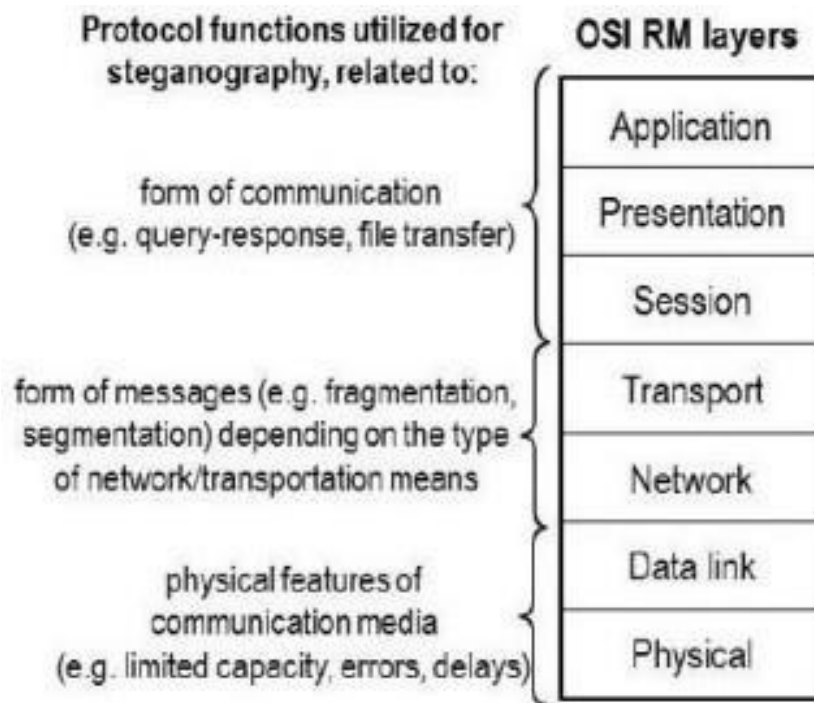
**Fig 3** Steganography logic

**Audio Steganography:** Audio steganography is the practise of incorporating information into audio files, such as mp3 sound files, wav files, or AU files, to convey a secret message that may take the shape of an image, audio file, text, or video. The techniques used for audio steganography are called hiding techniques, and the most common ones are insertion-based, substitution-based, and generation-based.

**Video Steganography :**that uses video frames to conceal data is known as video steganography. Because so much data that is transmitted over the internet is in the video format, this type is the one that is most commonly used. Along with the least significant bit (LSB) method, masking-filtering is another technique used in video steganography to conceal the hidden message. These both techniques are used for 24bit images ..

**Network Steganography:** In this type of steganography ,the secret information is hidden within the network .This method is more sophisticated in terms of usage of hiding the information in the header or packet of protocols depending on which method is being used in network steganography .In today's scenario people exchange information over social media like Face book ,Whatsapp , Skype , voice callingor video calling so there are numerous methods to hide the information over the network.

#### PROTOCOL FUNCTION USED BY IMAGE STEGANOGRAPHY



**Fig 4 Protocol function utilized for steganography**

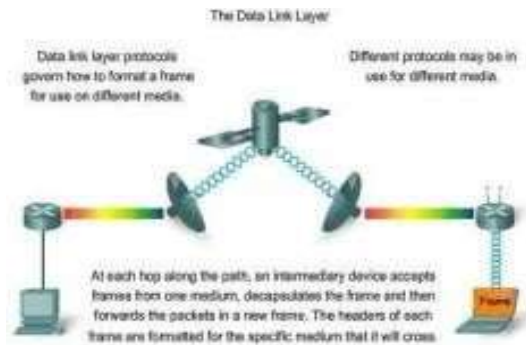
The various OSI RM layers that are related to steganography. since it is mainly a data transmission between two parties

## 1.9 OSI RMLAYERS THAT TAKE PART IN IMAGESTEGANOGRAPHY

### Datalink layer

In Layer 2 is the datalink layer in the Open Systems Interconnection (OSI) architecture model for a collection of communications protocols. Before being transferred, data bits were sorted, decoded, and encoded in this layer. The altered data bits are transmitted between two nearby nodes as frames. similar WAN or LAN.

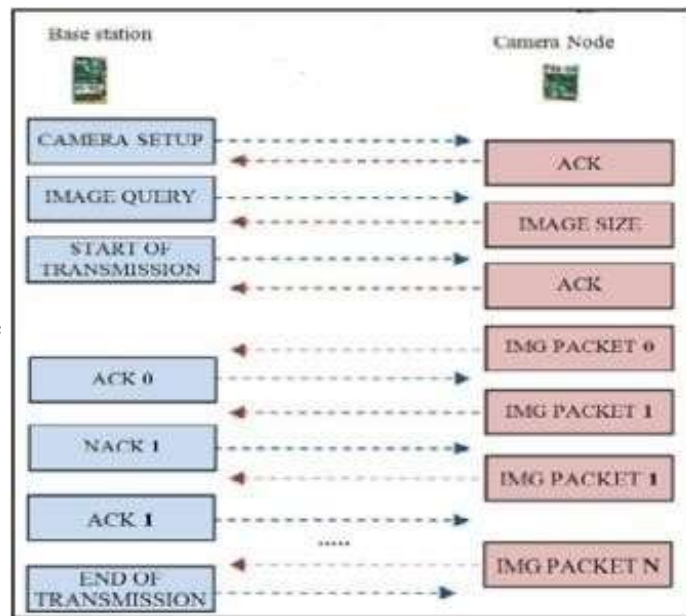
Fig 5 The Data link layer protocol



### Application layer

The application layer of a communications network defines the common communications protocols and host interface techniques. It is also referred to as the abstraction layer. Application layer abstraction is used by the Internet Protocol Suite (TCP/IP) and the OSI model, which are both widely accepted models of computer networking.

Fig 6 Application layer

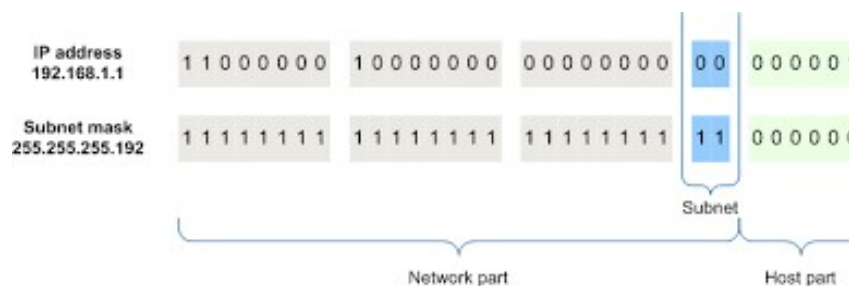


## SUBNETS-NETWORK LAYER

At Layer 3 of the OSI Model, or the IP layer, subnet operates. You can build smaller networks within of a bigger overall network using subnets.

Network ID and Host ID are the two parts of a typical IPv4 address. The first two segments of a class B IP address (172.16.1.10) are the Network ID (172.16), and the next two segments are the Host ID (1.10). When a request for information is made via the internet, it is first sent to the network and then, using the host ID, to the host. Each network will only have one of these numbers. To specify the subnet the host is situated in when building a subnet, you utilise a portion of the host ID. Using the same illustration,

the 1 in 172.16.1.10 would identify the subnet. In binary, 255.255.255.0 is 11111111.11111111.11111111.00000000 (twenty-four leading ones). The shorthand of using a "/" followed by a number is known as CIDR (Classless Inter-Domain Routing) notation and is very common in IPv4 networking.



**Fig 7-Subnet Masking**

## MAC ADDRESS RESOLUTION PROTOCOL-LINK LAYER

Address In a local area network, the Address Resolution Protocol (ARP) converts a dynamic IP address to a fixed physical machine address (LAN). A media access control (MAC) address is another name for the actual machine address. ARP's primary function is to convert 32-bit addresses to 48-bit addresses and the other way around. This is required because MAC addresses are 48 bits but IP addresses in IP version 4 (IPv4) are 32 bits.

ARP operates in the Open Systems Interconnection model's Layers 2 and 3. (OSI MODEL). The data link layer, or Layer 2, of the OSI model contains the MAC address. On Layer 3, the network layer, the IP address is present.

How ARP operates :

When a new computer joins a LAN, a special IP address is given to it for use in identification and communication. A gateway will ask the ARP software to discover a MAC address that matches the IP address when an incoming packet with a host machine's LAN address is received by the gateway. Each IP address is kept in a table called the ARP cache along with its accompanying MAC address.

**ARP:** ARP: The abbreviation (Address Resolution Protocol). It is in charge of using a known IP address to determine a host's hardware address. Three fundamental ARP phrases exist.

The following key words are connected to ARP:

1.Reverse ARP

2.Proxy ARP

3. Reverse ARP

1. ARP Cache: The MAC address is sent to the source by the ARP once it has been resolved and is then stored there in a table for future use. The MAC address from the table can be used in subsequent communications.

2. ARP Cache Timeout: It specifies how long the MAC address can stay in the ARP cache.

3. An ARP request is simply sending a packet across the network to see if we were able to find the target MAC address.

The physical address of the sender.

1. The IP address of the sender.

2. The physical address of the receiver is FF:FF:FF:FF:FF:FF or 1's.

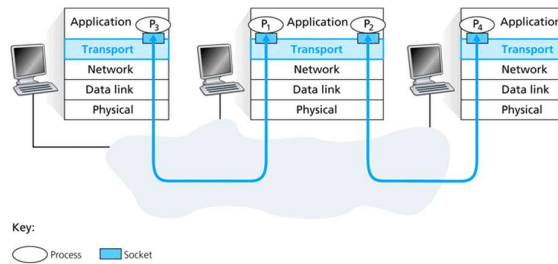
3. The IP address of the receiver

2. **ARP response/reply:** It is the MAC address response that the source receives from the destination which aids in further communication of the data.

## **SOCKET PROGRAMMING -TRANSPORT LAYER**

- A socket serves as the interface for communication between a process (application) and the transport layer. Every process has the ability to use a lot of sockets. Demultiplexing is the process by which a receiving machine's transport layer transfers segments from the network layer to the appropriate socket. Multiplexing is the process of combining information-packed segments and sending them to the network layer. Every time a communications channel is shared, multiplexing and demultiplexing are required.





**Fig 8 Transport layer**

## MERITS

- It is difficult to detect only the intended receiver can able to detect.
- It provides secured data transmission
- If we use large number of softwares, it can be done faster
- The advantage of steganography is that messages do not send consideration to themselves. Clearly detectable encrypted message no matter how tough will stimulate suspicion, and may in themselves be compromising in countries where encryption is illegitimate
- In steganography, cryptography secures the contents of a message, steganography can be said to secure both messages and connecting parties

## DEMERITS

- Large number of data and large file size which is considered to be disadvantage
- Image may become distorted. While compressing images such as JPEG, secret information may be lost. When steganography is implemented properly, it can be difficult to detect, but not impossible

## 1.10 SOCKET PROGRAMMING IN JAVA

Java Applications operating on various JREs communicate with one another using Java Socket programming. Programming using Java Sockets can be connection-oriented or connection-less. For connection-oriented socket programming, DatagramSocket and DatagramPacket classes are used, while Socket and ServerSocket are used for connection-less socket programming. Using socket programming, two nodes on a network can connect and communicate with one another. The other socket (node) reaches out to the first socket to establish a connection while the first socket listens on a certain port at an IP. While the client attempts to contact the server, the server creates the socket

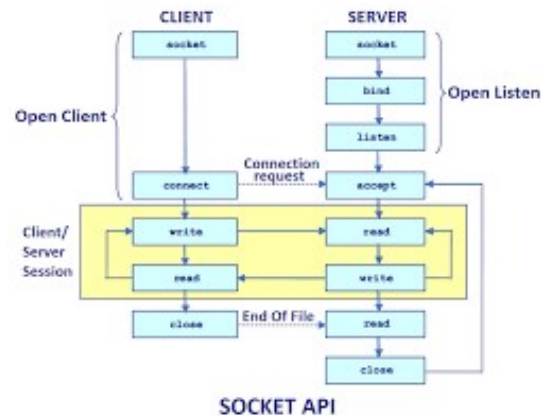
**Creating Client:** To create the client application, we need to create the instance of Socket class. Here, we need to pass the IP address or hostname of the Server and a port number. Here, we are using "localhost" because our server is running on same system.

```
Socket s=new Socket("localhost",XXXX);
```

## Creating Server:

To create the server application, we need to create the instance of ServerSocket class. Here, we are using XXXX port number for the communication between the client and server. You may also choose any other port number. The accept() method waits for the client. If clients connects with the given port number, it returns an instance of Socket.

1. ServerSocket ss=new ServerSocket(XXXX);
2. Socket s=ss.accept();//establishes connection and waits for the client



**Fig 9 state diagram of client and server**

Method	Description
1) public Socket accept()	returns the socket and establish a connection between server and client.
2) public synchronized void close()	closes the server socket

**Table 1. Client Socket class methods**

Method	Description
1) public InputStream getInputStream()	returns the InputStream attached with this socket.
2) public OutputStream getOutputStream()	returns the OutputStream attached with this socket.
3) public synchronized void close()	closes this socket

**Table 2 Server Socket class methods**

## **CHAPTER TWO**

### **LITERATURE REVIEW**

#### **2.1 OVERVIEW OF THE EXISTING RELATED WORK**

Many researchers have done research on the topic of internet data transfer security protection. Some of the work is seen below:

P. Marwaha and others, suggested that the most widely utilised methods are steganography and cryptography. Although both of these methods offer some data security, neither one is by itself secure enough to allow for information transfer via an insecure communication channel and is vulnerable to intrusion assaults.

Using DWT, Amitava Nag et al. describe a brand-new method for image steganography. Before embedding, the secret messages/picture are first encoded using Huffman using a 2D Discrete Wavelet Transform on a grey level cover image of size  $M \times N$ . The high frequency coefficients produced by DWT are then incorporated with each bit of the secret message or image's Huffman code.

#### **AN OVERVIEW OF ORIGIN OF STEGANOGRAPHY**

Herodotus' work can be used to trace the origins of steganography: The first Greek historian, Herodotus, describes a nobleman named Histaeus who needed to contact his son-in-law in Greece in Histories. One of his most dependable slaves had his skull tattooed with the inscription after it was shaved. The slave was sent with the secret message when his hair grew back.

The first printed work on cryptology was written by Johannes Trithemius. He created a cypher for Steganography in which each letter was a word pulled from a series of columns. The set of words that followed would constitute a valid prayer.

#### **2.2 AN OVERVIEW OF MODERN STEGANOGRAPHY**

In current steganography, data is concealed in the cover object, such as photos, audio/video, text, or protocol, without the data being changed into a different form. This is a change from earlier steganography approaches that involved changing the data into a different form.

Many digital file formats can act as cover objects for concealed data, but the one that has the highest redundancy—i.e., bits that offer precision much beyond what is required for the object's use and display—is the preferred one (Morkel, Eloff, & Olivier, 2005).

## CHAPTER THREE

### METHODOLOGY

#### 3.1 LEAST SIGNIFICANT BIT (LSB) ALGORITHM

The Least Significant Bit (LSB) algorithm adjusts the least significant bit pixels of the image which in this case is the carrier file. In this algorithm, the bit insertion depends on the number of bits in the image. For instance, in an 8bit image, the LSB of each byte of the image will be changed to the bit of the secret message. In 24bit image, the RGB (Red, Green, Blue) color components are substituted accordingly with the MSB (Most Significant Bit) of the secret data.

When used with a BMP image, LSB algorithm is very effective due to the lossless ability of bitmap image.

#### HIDING A TEXT WITHIN A PICTURE

##### Least Significant Bit (LSB)

- One of most common techniques
- Alters LSB of each pixel (1 bit out of 24 or 1 out of 8 for gray scale)
- Uses the concept of parity, i.e., even numbers in binary end in 0, odd one end in 1
- Easiest to implement: hiding bitmaps in a color picture
- Hiding ASCII code, one letter at a time

##### Example:

To hide letter C, which ASCII code is 67 and binary equivalent number is 1000011 in a gray scale file:

##### Original:

0100110	0100111	0100111	0100111	0101000	0101000
1	0	0	1	0	0
1	0	0	0	0	1
0100111					
1					
1					

##### Encoded:

0100110	0100111	0100111	0100111	0101000	0101000
1	0	0	1	0	1
0100111					
1					

### ***Encoding a message***

The function encode() creates an image with message written in big black letters across it; it then invokes encodePicture() to hide the image with the message in the given picture

## **DESIGN AND ANALYSIS OF SYSTEM**

### ***3.1.1 ANALYSIS :***

I research the interrelated sets of text, algorithm, key, and other components that make up the system. The following phases would be included in various system analysis methodologies that follow the waterfall paradigm.

- The creation of a feasibility study, which involves figuring out whether a project is possible from an organisational, societal, and economic standpoint.
- Carrying out fact-finding procedures intended to discover the needs of the system's end users.

estimating the system's intended use, the end user's operating style (in terms of their overall familiarity with utilising computer hardware and software), and other factors. Analysis of the suggested system is conducted using an interaction and structural model of systems.

### ***3.1.2 DESIGN:***

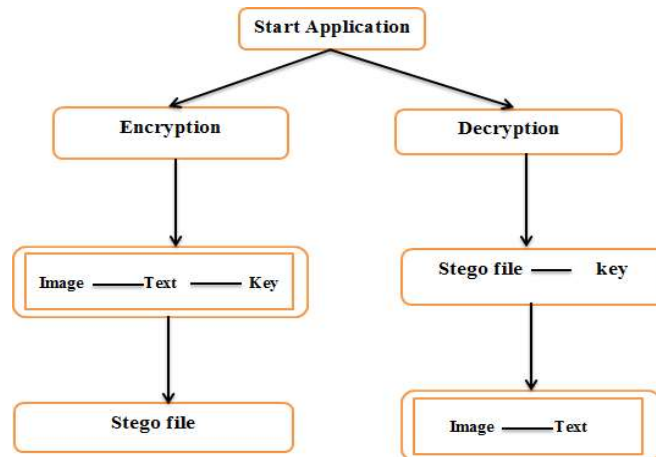
This is the process of designing a new system, one to replace an existing system, or one to enhance an existing system. However, before this planning can be completed, we must fully comprehend the existing system and identify how computers can be utilised to improve its performance. I added client and server use case diagrams for the project. Use case diagrams are behavioural diagrams that show a series of tasks that a system or systems should or could complete in coordination with one or more external users of the system. Each use case should produce a tangible benefit for the system's actors or other stakeholders.

## ANALYSIS OF THE PROPOSED SYSTEM

The suggested system is concerned with information and methods for protecting it. Steganography is one of the information security techniques used. The internet's rapid advancement in data transfer made it simpler to send data accurately and quickly to its destination. One of the key components of information technology and communication is information security. The art and science of invisible communication is steganography. The way the message's existence can be concealed is through steganography. This is done by obscuring the existence of the sent information by obscuring it in other information.

**Fig 10 Proposed System**

### PROPOSED ALGORITHM



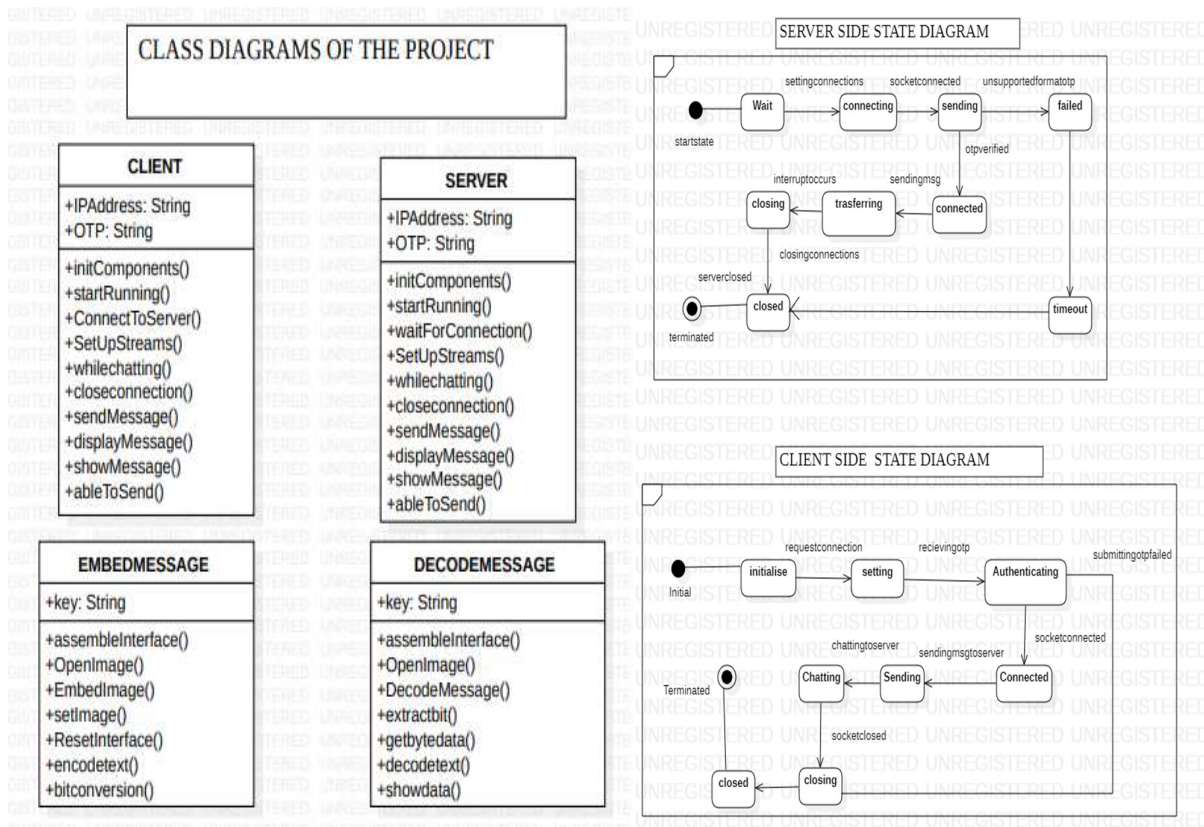
### ALGORITHM TO EMBED TEXT MESSAGE:

- Step 1: Read the cover image and text message which is to be hidden in the cover image.
- Step 2: Convert text message in binary format.
- Step 3: Calculate LSB of each pixels of cover image.
- Step 4: Replace LSB of cover image with each bit of secret message one by one.
- Step 5: Write stego image
- Step 6: Calculate the Mean Square Error (MSE) and the Peak signal to noise ratio (PSNR) of the stego image

### ALGORITHM TO DECODE TEXT MESSAGE:

- Step 1: Read the stego image.
- Step 2: Calculate LSB of each pixel of stego image.
- Step 3: Retrieve bits and convert each 8 bit into character.

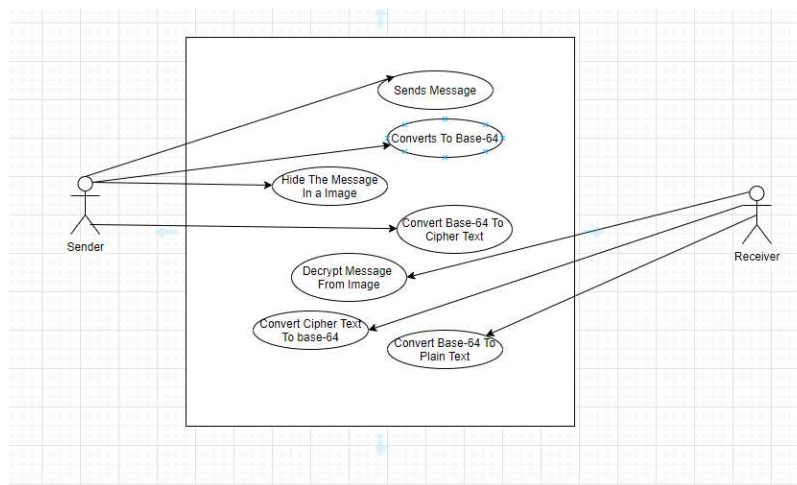
### 3.3 CLASS AND STATE MODEL



**Fig 11 – Class Diagram & State Diagram**

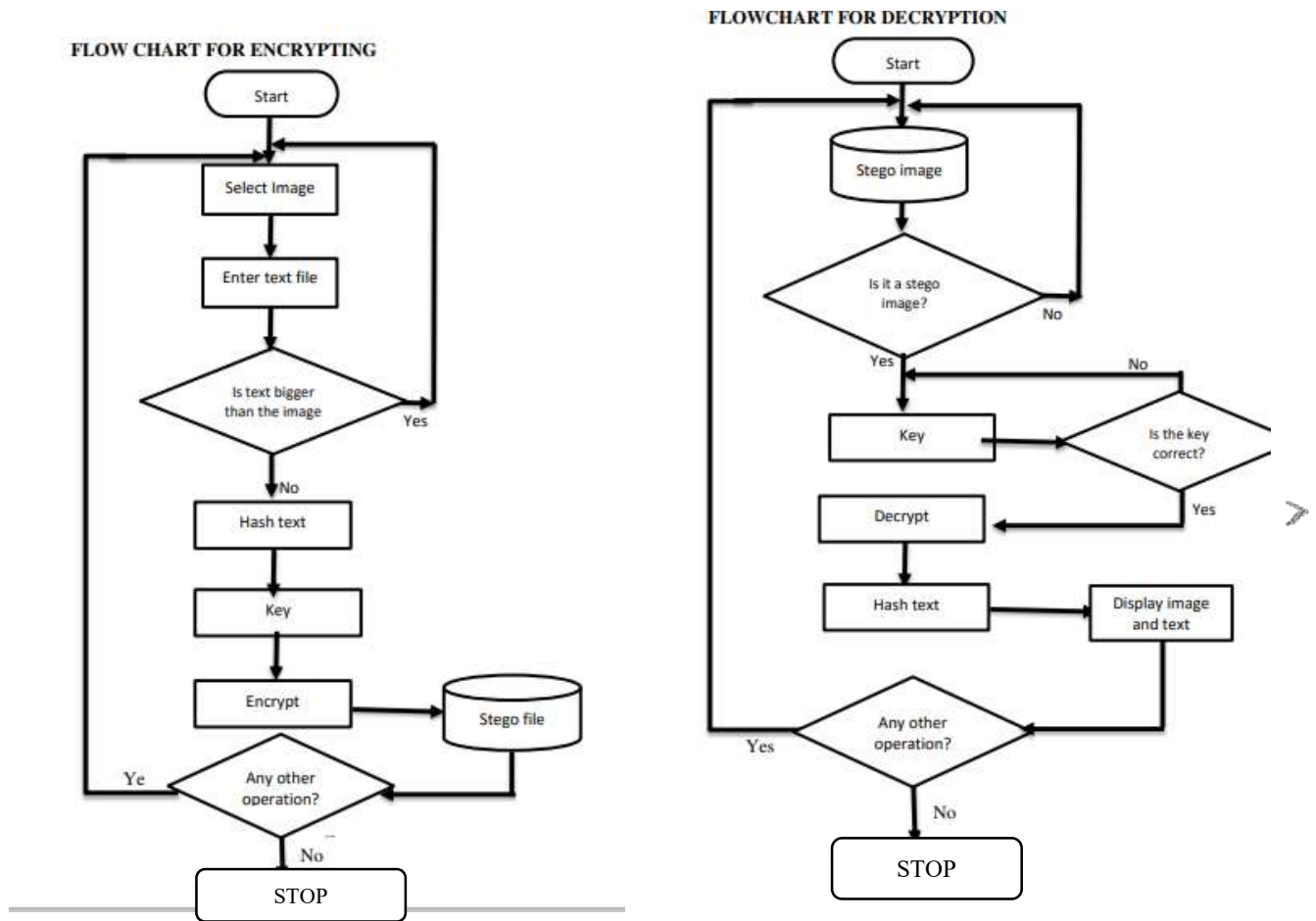
#### 3.3.1 USE CASE DIAGRAM

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved

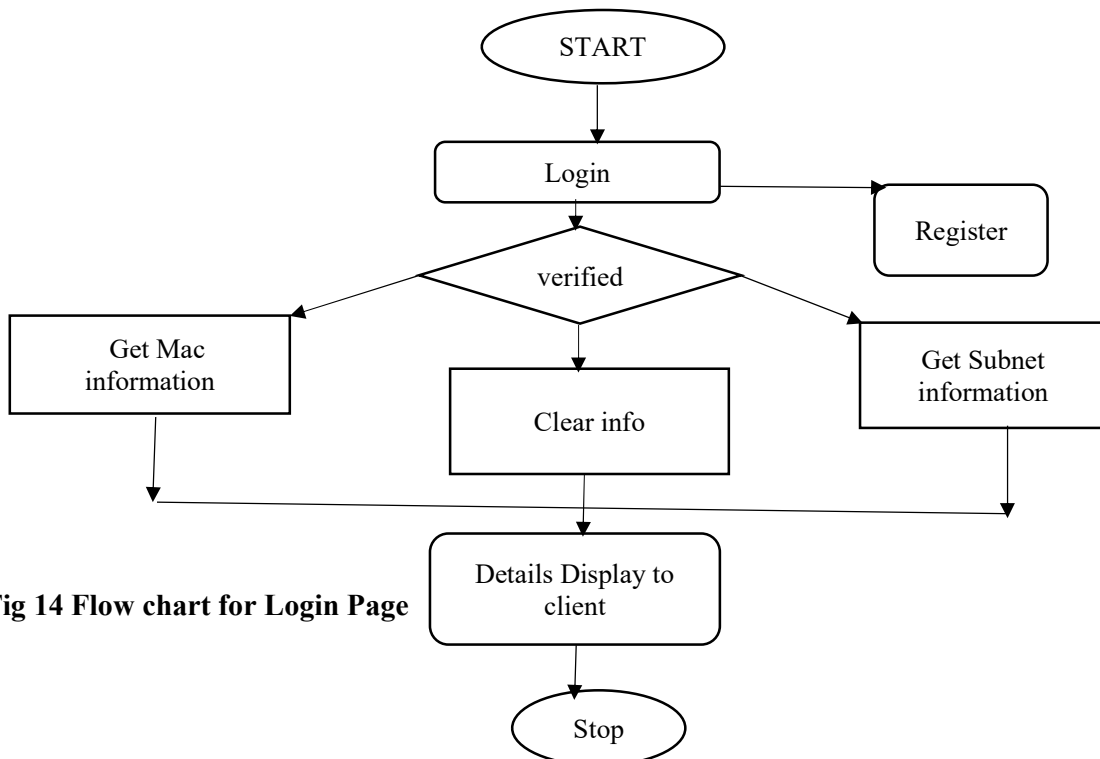


**Fig 12 Use case Diagram**

### 3.3.1 FLOW CHART DIAGRAMS OF THE PROJECT



**Fig 13 Flow chart diagrams for Encryption and Decryption**



**Fig 14 Flow chart for Login Page**



## CHAPTER FOUR

### IMPLEMENTATION AND RESULTS

#### 4.1.1 SOURCE CODE OF THE PROJECT

##### Loginpage.java

```
package jdbcpkg;
import java.awt.*;
import java.util.*;
import javax.swing.*;
import java.awt.event.*;
import java.awt.image.BufferedImage;
import java.io.*;
import java.sql.*;

class IDandPasswords {
    HashMap<String,String> logininfo = new HashMap<String,String>();
    String user;
    String pass;
    String regno;
    String grade;
    String sbranch;
    IDandPasswords(){
        try{
            Class.forName("oracle.jdbc.driver.OracleDriver");
            Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","system","12345678");
            Statement stmt=con.createStatement();
            ResultSet rs=stmt.executeQuery("select * from Student");
            while(rs.next())
            {
                System.out.println("REGNO : "+rs.getString(1)+" STUDENTNAME : "+
rs.getString(2)+" PASSWORD : "+rs.getString(3)+" CGPA : "+rs.getString(4)+" BRANCH: "+rs.getString(5));
                regno=rs.getString(1);
                user=rs.getString(2);
                pass=rs.getString(3);
                grade=rs.getString(4);
                sbranch=rs.getString(5);
                logininfo.put(user,pass);
            }
            System.out.println(logininfo);
        } catch (Exception e){ System.out.println(e);}
    }

    public HashMap getLoginInfo(){
        return logininfo;
    }
}

class Login implements ActionListener{
    JFrame frame = new JFrame();
    JButton loginButton = new JButton("LOGIN");
    JButton resetButton = new JButton("RESET");
    JButton register=new JButton("REGISTER");
    JTextField userIDField = new JTextField(20);
    JPasswordField userPasswordField = new JPasswordField(18);
    JLabel userIDLabel = new JLabel("USERID:");
    JLabel userPasswordLabel = new JLabel("PASSWORD:");
    JLabel messageLabel = new JLabel();
    JLabel loginform=new JLabel("LOGIN FORM");
    HashMap<String,String> logininfo = new HashMap<String,String>();
    BufferedImage img;
    JFrame fm;
    JTextArea distxt=new JTextArea(15,15);
    Login(HashMap<String,String> loginInfoOriginal) {
```

```

logininfo = loginInfoOriginal;
userIDLabel.setBounds(50,100,75,25);
userPasswordLabel.setBounds(50,150,75,25);
loginform.setBounds(125,250,250,35);
loginform.setFont(new Font(null,Font.CENTER_BASELINE,40));
messageLabel.setBounds(125,250,250,35);
messageLabel.setFont(new Font(null,Font.ITALIC,30));
userIDField.setBounds(125,100,200,25);
userPasswordField.setBounds(125,150,200,25);
loginButton.setBounds(125,200,100,25);
loginButton.setFocusable(false);
loginButton.addActionListener(this);
resetButton.setBounds(225,200,100,25);
resetButton.setFocusable(false);
resetButton.addActionListener(this);
register.setBounds(225,200,100,25);
register.setFocusable(false);
register.addActionListener(this);
frame.add(loginform);
frame.add(userIDLabel);
frame.add(userIDField);
frame.add(userPasswordLabel);
frame.add(userPasswordField);
frame.add(messageLabel);
frame.add(loginButton);
frame.add(resetButton);
frame.add(register);
frame.add(distxt);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setSize(300,400);
frame.setLayout(new FlowLayout());
frame.setVisible(true);
frame.setResizable(false);
}
public void actionPerformed(ActionEvent e) {
    if(e.getSource()==resetButton) {
        userIDField.setText("");
        userPasswordField.setText("");
    }
    if(e.getSource()==loginButton) {
        String userID = userIDField.getText();
        String password = String.valueOf(userPasswordField.getPassword());
        if(logininfo.containsKey(userID)) {
            if(logininfo.get(userID).equals(password)) {
                messageLabel.setForeground(Color.green);
                messageLabel.setText("Login successful");
                frame.dispose();
                JOptionPane.showInputDialog("Hello "+userID+" !!\nYou are Logged
in Successfully..\n Enter ok");
                JOptionPane.showInputDialog(" YOUR REQUESTED IP ADDRESS is : 172.22.62.202 ")
;
                new Netinfo();
            }
            else {
                messageLabel.setForeground(Color.red);
                messageLabel.setText("Wrong password");
            }
        }
        else {
            messageLabel.setForeground(Color.red);
            messageLabel.setText("username not found");
        }
    }
    if(e.getSource()==register) {

```

```

        fm=new JFrame();
        JTextField t1=new JTextField(25);
        JTextField t2=new JTextField(25);
        JTextField t3=new JTextField(30);
        JTextField t4=new JTextField(30);
        JTextField t5=new JTextField(30);
        JTextArea txt=new JTextArea(8,8);
        JLabel j0=new JLabel("REGISTRATION FORM");
        JLabel j1=new JLabel("REGISTRATION NUMBER");
        JLabel j2=new JLabel("STUDENT NAME ");
        JLabel j3=new JLabel(" CREATE PASSWORD");
        JLabel j4=new JLabel("ENTER GPA");
        JLabel j5=new JLabel("ENTER BRANCH");
        JButton submit =new JButton("SUBMIT");
        submit.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae) {
            try {
                String roll,name,pwd,cgpa,branch;
                Connection
con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","system","12345678");
                PreparedStatement ps=con.prepareStatement("insert into student values(?,?,?,?,?)");
                roll=t1.getText();
                name=t2.getText();
                pwd=t3.getText();
                cgpa=t4.getText();
                branch=t5.getText();
                ps.setString(1, roll);
                ps.setString(2, name);
                ps.setString(3, pwd);
                ps.setString(4, cgpa);
                ps.setString(5, branch);
                ResultSet rs2=ps.executeQuery();
                System.out.println("RECORD INSERTED SUCCESSFULLY..\n");
                txt.setText("REGISTRATION COMPLETED SUCESSFULLY..");
            }catch(Exception eM)
            {
                eM.printStackTrace();
            }
        }
    });

    j0.setBounds(125,250,250,35);
    j0.setFont(new Font(null,Font.CENTER_BASELINE,40));
    fm.add(j0);
    fm.add(j1);
    fm.add(t1);
    fm.add(j2);
    fm.add(t2);
    fm.add(j3);
    fm.add(t3);
    fm.add(j4);
    fm.add(t4);
    fm.add(j5);
    fm.add(t5);
    fm.add(submit);
    fm.add(txt);
    fm.setBackground(Color.cyan);
    fm.setForeground(Color.red);
    fm.setSize(500,500);
    fm.setLayout(new FlowLayout());
    fm.setVisible(true);
    fm.setResizable(true);
    fm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}
}
}

```

```

public class LoginPage {
public static void main(String[] args) {
    IDandPasswords idandPasswords = new IDandPasswords();
    Login loginPage = new Login(idandPasswords.getLoginInfo());

}

}

```

### **Netinfo.java**

```

package jdbcpkg;
import java.awt.*;
import java.awt.event.*;
import java.net.*;
import javax.swing.*;
import java.util.Scanner;
public class Netinfo {
JFrame frame=new JFrame("NETWORK INFORMATION");
JTextField t1=new JTextField(20);
JTextField t2=new JTextField(20);
JLabel l1=new JLabel("ENTER IP ADDRESS :");
JLabel l2=new JLabel("ENTER NUM OF ADDRESS IN SUBNET:");
JButton display=new JButton("subnetdetails");
JButton clear=new JButton("clear");
JButton map=new JButton("MAC");
JTextArea txt=new JTextArea(15,15);
Netinfo(){
    map.setBounds(225,200,100,25);
    map.setFocusable(false);
    map.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
    try
    {String ipaddr = t1.getText();
InetAddress address = InetAddress.getByIp(ipaddr);
txt.append("\n\n\nDISPLAYING MAC INFORMATION USING MAC ADDRESS RESOLUTION PROTOCOL\n");
txt.append("IP ADDRESS :"+address+"\n");
NetworkInterface ni = NetworkInterface.getByInetAddress(address);
if (ni!=null)
{ byte[] mac = ni.getHardwareAddress();
if (mac != null)
{ txt.append("MAC ADDRESS : ");
for (int i=0; i < mac.length; i++)
{
txt.append((String.format("%02X%s", mac[i], (i < mac.length - 1) ? "-" : "")));
}
}
}
else
{
txt.append("\nAddress doesn't exist or is not accessible/");

}
}
}
else
{
txt.append("\nNetwork Interface for the specified address is not found");
}
}
}
catch(UnknownHostException | SocketException ea)
{
}}
});
clear.setBounds(225,200,100,25);
clear.setFocusable(false);

```

```

        clear.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e) {
            txt.setText(" ");
        }});
display.setBounds(225,200,100,25);
display.setFocusable(false);
display.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent e) {
        txt.append("\nDISPLAYING SUBNET INFORMATION\n");
        txt.append("Ip address: ");
        String ip=t1.getText();
        txt.append(ip+"\n");
        String split_ip[] = ip.split("\\."); //Split the string after every .
        String split_bip[] = new String[4]; //split binary ip
        String bip = "";
        for(int i=0;i<4;i++){
            split_bip[i] = appendZeros(Integer.toBinaryString(Integer.parseInt(split_ip[i])));
            bip += split_bip[i];
        }
        txt.append("Binary Format "+bip+"\n");
        String a=t2.getText();
        int n=Integer.parseInt(a);
        //Calculation of mask
        int bits = (int)Math.ceil(Math.log(n)/Math.log(2));
        int mask = 32-bits;
        txt.append("Subnet mask = "+mask+"\n");
        int fbip[] = new int[32];
        for(int i=0; i<32;i++) fbip[i] = (int)bip.charAt(i)-48; //convert cahracter 0,1 to integer 0,1
        for(int i=31;i>31-bits;i--)//Get first address by ANDing last n bits with 0
            fbip[i] &= 0;
        String fip[] = {"", "", "", ""};
        for(int i=0;i<32;i++)
            fip[i/8] = new String(fip[i/8]+fbip[i]);
        txt.append("\nNetwork address is "+" :");
        for(int i=0;i<4;i++){
            txt.append(" "+Integer.parseInt(fip[i],2)+"");
            if(i!=3)
                System.out.print(" ");
            txt.append(" ");
        }txt.append(" ");
        int lbip[] = new int[32];
        for(int i=0; i<32;i++) lbip[i] = (int)bip.charAt(i)-48; //convert character 0,1 to integer 0,1
        for(int i=31;i>31-bits;i--)//Get last address by ORing last n bits with 1
            lbip[i] |= 1;
        String lip[] = {"", "", "", ""};
        for(int i=0;i<32;i++)
            lip[i/8] = new String(lip[i/8]+lbip[i]);
        txt.append("\nBroadcast address is = ");
        for(int i=0;i<4;i++){
            txt.append(" "+Integer.parseInt(lip[i],2)+"");
            if(i!=3)
                txt.append(" ");
        }txt.append(" ");
    }
});
frame.add(l1);
frame.add(t1);
frame.add(l2);
frame.add(t2);
frame.add(display);
frame.add(clear);
frame.add(map);
frame.add(txt);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

```

```

frame.setSize(500,600);
frame.setLayout(new FlowLayout());
frame.setVisible(true);
frame.setResizable(false);
    }
    public String appendZeros(String s){
    String temp = new String("00000000");
    return temp.substring(s.length()+ s;
    }
    }

```

## **Serv.java**

```

package jdbcpkg;
import java.awt.*;
import java.awt.event.*;
import java.awt.image.BufferedImage;
import java.io.*;
import java.lang.reflect.InvocationTargetException;
import java.net.ServerSocket;
import javax.swing.*;
import java.net.*;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.imageio.ImageIO;

```

```

public class Serv extends JFrame{
private JLabel statuslabel,chatlabel,instruction,instruction1;
private JTextArea statusarea;
private JPanel chatbox;
private JScrollPane statusscroll,scroll;private JButton
Embed,Send,Decode; private ServerSocket server;
private Socket connection; private
ObjectInputStream input; private
ObjectOutputStream output;private
JCheckBox []checks;

```

```

private BufferedImage[] images;
private ButtonGroup grp; public
BufferedImage Bufim;public Serv
sser;
public JLabel setlabel;
private int JC;
private String security=null;

public Serv(){
    super("Server");
    Bufim=null;
    sser=this;
    JC=-1;
    this.setVisible(true);
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    this.setBounds(10,20,700,700); this.setResizable(false);
    this.setLayout(null); this.getContentPane().setBackground(Color.DARK_GRAY);

    this.addWindowListener(new WindowAdapter(){

        @Override
        public void windowClosing(WindowEvent e) {
            closeConnection();
        }
    });

    initComponents();
}
private void initComponents(){ statuslabel=new
JLabel("STATUS :");
    statuslabel.setForeground(Color.WHITE);
    statuslabel.setBounds(50,5,70,50); add(statuslabel);

    statusarea=new JTextArea(); statusscroll=new
JScrollPane(statusarea);
    statusscroll.setBounds(115,5,300,100);
    add(statusscroll);

    instruction=new JLabel("Please 'EMBED' to prepare message !");
    instruction.setForeground(Color.WHITE); instruction.setBounds(420,5,250,50);
    add(instruction);

    instruction1=new JLabel("Please select a message to 'READ' !");instruction1.setForeground(Color.WHITE);
    instruction1.setBounds(420,60,270,50);
    add(instruction1);

    chatlabel=new JLabel("CHAT WINDOW :");
    chatlabel.setForeground(Color.WHITE);
    chatlabel.setBounds(50,110,100,50); add(chatlabel);

    checks=new JCheckBox[100]; grp=new
ButtonGroup(); images=new
BufferedImage[100];

    Embed=new JButton("ENCODE");
    Embed.setBounds(50,610,100,40); Embed.addActionListener(new
ActionListener(){

```

```

        @Override
        public void actionPerformed(ActionEvent e) {
            new EmbedMessage(sser,security);
        }
    });
    add(Embed);

    setlabel=new JLabel("No image is set!");
    setlabel.setForeground(Color.WHITE);
    setlabel.setBounds(150,610,150,40); add(setlabel);

    Decode=new JButton("DECODE");
    Decode.setBounds(550,610,100,40);
    ////////// Decode.setEnabled(false);
    Decode.addActionListener(new ActionListener() { @Override
        public void actionPerformed(ActionEvent e) {BufferedImage
            BB=null;
            for(int i=0;i<=JC;i++){
                if(checks[i].isSelected()) {BB=images[i];
                    break;
                }
            }
            new DecodeMessage(BB);
        }
    });
    add(Decode);

    chatbox=new JPanel();
    BorderLayout layout=new BorderLayout(chatbox,BoxLayout.Y_AXIS); chatbox.setBackground(Color.black);
    chatbox.setBorder(BorderFactory.createLineBorder(Color.CYAN));chatbox.setLayout(layout);
    scroll = new JScrollPane(chatbox);
    scroll.setBounds(50,150,600,450); add(scroll);
    ////////// grp=new
    ButtonGroup(); Send=new
    JButton("SEND");
    Send.setBounds(300,610,100,40); Send.setEnabled(false);
    Send.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) { SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() { BufferedImage
                IM=Bufim;if(IM==null){
                    return;
                }
                Decode.setEnabled(true); JPanel
                Panel=new JPanel();
                Panel.setLayout(new FlowLayout(FlowLayout.RIGHT));
                Panel.setBackground(Color.white);

                JCheckBox chk=new JCheckBox("SERVER");
                chk.setSize(100, 50); checks[++JC]=chk;
                grp.add(checks[JC]);
            }
        });
    });

```



```

        JLabel lb=new JLabel();
        lb.setSize(150,150);
        lb.setIcon(new ImageIcon(IM.getScaledInstance(lb.getWidth(),
        lb.getHeight(),
        Image.SCALE_SMOOTH)));

        images[Jc]=IM;

        Panel.add(checks[Jc]);Panel.add(lb);
        chatbox.add(Panel);
        chatbox.add(Box.createVerticalStrut(20));sser.validate();
        sser.repaint();
        sendMessage(IM);

    }

    });

    }

    });
    add(Send);

    sser.validate();
    sser.repaint();

}

public static void main(String[] args) throws InterruptedException, InvocationTargetException {Serv ser=new Serv();
    ser.startRunning();

}

public void startRunning(){////////////////////
    try{
        server = new ServerSocket(7777,100);
        while(true){
            try{
                waitForConnection();
                setupStreams();
                whileChatting();
            }catch(Exception eofException){ showMessage("\nServer ended the
                connection!");
            } finally{
                closeConnection();
            }
        }
    } catch (IOException ioException){ ioException.printStackTrace();
    }

}

private void waitForConnection() throws IOException{////////////////////showMessage("\nWaiting for someone to
connect...");
    connection = server.accept();
    showMessage("\nNow connected to " + connection.getInetAddress().getHostName());
}

private void setupStreams() throws IOException{////////////////////output = new
    ObjectOutputStream(connection.getOutputStream());
    output.flush();

    input = new ObjectInputStream(connection.getInputStream());

    showMessage("\nStreams are now setup");}

```

```

private void whileChatting() throws IOException { //////////////////////////////////////////////////chatbox.removeAll();
    this.repaint();
    this.validate();
    /*String s=JOptionPane.showInputDialog(this,"Hey Server please set the OTP
    for
Client!");

    if(s.equals("")){
        s="12345";
    }*/
    double dd;
    while((dd=Math.random())<0.1){} Integer
    xx=(int)(dd*100000); String
    s=xx.toString(); security=s;
    String message = " You are now connected!\n Your OTP : "+s;
    System.out.println("SENDED OTP:"+s);
    statusarea.append("\n\nSENDED OTP:"+s);
    sendMessage(message);
    try {
        String ss=(String)input.readObject();
        if(!ss.equals(security)){ sendMessage("Incorrect OTP
        !");return;
        }
        else sendMessage("Verification Successfull !");
    } catch (ClassNotFoundException ex) { Logger.getLogger(Clin.class.getName()).log(Level.SEVERE,
        null, ex);
    }
    ableToSend(true); Object
    mess=null;
    do{
        try{
            mess =input.readObject();
            if(mess instanceof String){
                if(!((String)mess).equals("CLOSE"))showMessage("\n" + mess);
            }
            else {
                ByteArrayInputStream bin=new ByteArrayInputStream((byte[]) mess);BufferedImage
                ms=ImageIO.read(bin);
                displayMessage(ms);
                bin.close();
            }
        } catch (Exception Ex){
            showMessage("\nThe user has sent an unknown object!");
        }
    } while(mess instanceof byte[] || (mess!=null && !((String)mess).equals("CLOSE")));
}

public void closeConnection() { //////////////////////////////////////////////////sendMessage("CLOSE");
    showMessage("\nClosing Connections...");
    ableToSend(false);
    try{
        output.close();
        input.close();
        connection.close();
    } catch (IOException ioException) {
        showMessage("\nError in closing the streams!");
    }
} private void sendMessage(Object message) { //////////////////////////////////////////////////

```

```

        ByteArrayOutputStream bos;
        if(message==null){
            return;
        }
        try{
            if(message instanceof String){ output.writeObject(message);
                output.flush();
            }else{
                BufferedImage bm=(BufferedImage)message; bos=new
                ByteArrayOutputStream(); ImageIO.write(bm, "png", bos
                );
                byte[] imageInByte = bos.toByteArray(); bos.flush();
                output.writeObject(imageInByte);
                output.flush(); bos.close();
            }
            setlabel.setText("No image is set!"); Bufim=null;
        } catch(IOException ioException){
            statusarea.append("\n Error : Cannot send message!, PLEASE RETRY");
        }
    }

    private void displayMessage(BufferedImage mes){ SwingUtilities.invokeLater(new
        Runnable(){
            @Override
            public void run() {
                if(mes==null){
                    return;
                }
                Decode.setEnabled(true);
                BufferedImage IM=mes; JPanel
                Panel=new JPanel();
                Panel.setLayout(new FlowLayout(FlowLayout.LEFT));
                Panel.setBackground(Color.white);

                JCheckBox chk=new JCheckBox("CLIENT");
                chk.setSize(100, 50); checks[++JC]=chk;
                grp.add(checks[JC]);

                JLabel lb=new JLabel();
                lb.setSize(150,150);
                lb.setIcon(new ImageIcon(IM.getScaledInstance(lb.getWidth(),
                lb.getHeight(),
                Image.SCALE_SMOOTH)));

                images[JC]=IM;

                Panel.add(lb); Panel.add(checks[JC]);

                chatbox.add(Panel); chatbox.add(Box.createVerticalStrut(20));
                sser.repaint();
                sser.validate();
            }
        });
    }

    private void showMessage(final String text){////////////////////////////////////SwingUtilities.invokeLater(

```

```

        new Runnable(){
            @Override
            public void run(){
                statusarea.append(text);
            }
        }
    );
}

private void ableToSend(final boolean tof){////////////////////////////////////SwingUtilities.invokeLater(
    new Runnable(){
        @Override
        public void run(){
            Send.setEnabled(tof);
        }
    }
);
}
}

```

## **Clin.java**

```

package jdbcpkg;
import java.awt.*;
import java.awt.event.*;
import java.awt.image.BufferedImage;
import java.io.*;
import java.lang.reflect.InvocationTargetException;
import java.net.ServerSocket;
import javax.swing.*;
import java.net.*;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.imageio.ImageIO;
import java.sql.*;
public class Clin extends JFrame{
    private JLabel statuslabel,chatlabel,instruction,instruction1;
    private JTextArea statusarea;
    private JPanel chatbox;
    private JScrollPane statusscroll; private
    JButton Embed,Send,Decode;private Socket
    connection; private ObjectInputStream input;
    private ObjectOutputStream output;private
    JCheckBox []checks; private
    BufferedImage[] images; private
    ButtonGroup grp;
    private String ServerIP; public
    BufferedImage Bufim;public Clin
    cli;
    public JLabel setlabel;
    private int JC;private String security=null;

```

```

public Clin(String host){
    super("Client");
    Bufim=null; cli=this;
    JC=-1;
    ServerIP=host;
    this.setVisible(true);
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    this.setBounds(720,20,700,700); this.setResizable(false);
    this.setLayout(null); this.getContentPane().setBackground(Color.DARK_GRAY);

    this.addWindowListener(new WindowAdapter(){

        @Override
        public void windowClosing(WindowEvent e) {
            closeConnection();
        }
    });

    initComponents();
}

private void initComponents(){ statuslabel=new
    JLabel("STATUS :");
    statuslabel.setForeground(Color.WHITE);
    statuslabel.setBounds(50,5,70,50); add(statuslabel);

    statusarea=new JTextArea(); statusscroll=new
    JScrollPane(statusarea);
    statusscroll.setBounds(115,5,300,100);
    add(statusscroll);

    instruction=new JLabel("Please 'EMBED' to prepare message !");
    instruction.setForeground(Color.WHITE); instruction.setBounds(420,5,250,50);
    add(instruction);

    instruction1=new JLabel("Please select a message to 'READ' !");
    instruction1.setForeground(Color.WHITE); instruction1.setBounds(420,60,270,50);
    add(instruction1);

    chatlabel=new JLabel("CHAT WINDOW :");
    chatlabel.setForeground(Color.WHITE);
    chatlabel.setBounds(50,110,100,50); add(chatlabel);

    checks=new JCheckBox[100]; grp=new
    ButtonGroup(); images=new
    BufferedImage[100];

    Embed=new JButton("ENCODE");
    add(Embed);
    Embed.setBounds(50,610,100,40);
    Embed.addActionListener(new ActionListener(){@Override
        public void actionPerformed(ActionEvent e) {
            new EmbedMessage(cli,security);
        }
    });
}

```

```

setlabel=new JLabel("No image is set!");
setlabel.setForeground(Color.WHITE);
setlabel.setBounds(150,610,150,40); add(setlabel);

Decode=new JButton("DECODE");
Decode.setBounds(550,610,100,40);

Decode.setEnabled(false); Decode.addActionListener(new
ActionListener(){
    @Override
    public void actionPerformed(ActionEvent e) {BufferedImage
        BB=null;
        for(int i=0;i<=JC;i++){
            if(checks[i].isSelected()){BB=images[i];
                break;
            }
        }
        new DecodeMessage(BB);
    }
});
add(Decode);

chatbox=new JPanel();
BoxLayout layout=new BoxLayout(chatbox,BoxLayout.Y_AXIS);
chatbox.setBackground(Color.BLACK);
chatbox.setBorder(BorderFactory.createLineBorder(Color.CYAN));chatbox.setLayout(layout);
JScrollPane scroll = new JScrollPane(chatbox);scroll.setBounds(50,150,600,450);
add(scroll);

grp=new ButtonGroup();

Send=new JButton("SEND");

Send.setEnabled(false); Send.setBounds(300,610,100,40);
Send.addActionListener(new ActionListener(){
    @Override
    public void actionPerformed(ActionEvent e) { SwingUtilities.invokeLater(new Runnable(){
        @Override
        public void run() { BufferedImage
            IM=Bufim;if(IM==null){
                return;
            }
            Decode.setEnabled(true); JPanel
            Panel=new JPanel();
            Panel.setLayout(new FlowLayout(FlowLayout.RIGHT));
            Panel.setBackground(Color.white);

            JCheckBox chk=new JCheckBox("CLIENT");
            chk.setSize(100, 50); checks[++JC]=chk;
            grp.add(checks[JC]);

            JLabel lb=new JLabel();
            lb.setSize(150,150);
        }
    });
}

```

```
lb.setIcon(new ImageIcon(IM.getScaledInstance(lb.getWidth(), lb.getHeight(), Image.SCALE_SMOOTH)));
```

```
images[JC]=IM;
```

```
Panel.add(checks[JC]);Panel.add(lb);
```

```
chatbox.add(Panel);
```

```
chatbox.add(Box.createVerticalStrut(20));cli.validate();
```

```
cli.repaint();
```

```
sendMessage(IM);
```

```
    }
```

```
  });
```

```
  }
```

```
});
```

```
add(Send);
```

```
cli.validate();
```

```
cli.repaint();
```

```
}
```

```
public static void main(String[] args) {
```

```
    //Clin clin=new Clin("172.22.11.186");Clin
```

```
    clin=new Clin("localhost");
```

```
    clin.startRunning();
```

```
}
```

```
public void startRunning(){////////////////////////////////////
```

```
    try{
```

```
        connectToServer();
```

```
        setupStreams();
```

```
        whileChatting();
```

```
    } catch (EOFException eofException){
```

```
        showMessage("\nClient terminated the connection");
```

```
    } catch (IOException ioException){
```

```
        ioException.printStackTrace();
```

```
    } finally{
```

```
        closeConnection();
```

```
    }
```

```
}
```

```
private void connectToServer() throws IOException{
```

```
    showMessage("\nAttempting connection...");
```

```
    connection = new Socket(InetAddress.getByName(ServerIP), 7777);
```

```
    showMessage("\nConnection Established! Connected to: " +
```

```
connection.getInetAddress().getHostName());
```

```
}
```

```
private void setupStreams() throws IOException {////////////////////////////////////output = new
```

```
ObjectOutputStream(connection.getOutputStream());
```

```
output.flush();
```

```
    input = new ObjectInputStream(connection.getInputStream());
```

```
    showMessage("\nStreams are now setup");
```

```
}
```

```
private void whileChatting() throws IOException {////////////////////////////////////
```

```

///////// ableToSend(true);
Object mess=null;
try {
    showMessage("\n" + (String)input.readObject()); security=JOptionPane.showInputDialog(this,"Hey Client please provide
you just recieved!");

    if(security==null){
        security="-";
    }
    sendMessage(security);
    String rs=(String)input.readObject(); showMessage("\n" +
((rs.equals("CLOSE"))?"":rs));System.out.println("OTP
VERIFIED\n");
} catch (ClassNotFoundException ex) { Logger.getLogger(Clin.class.getName()).log(Level.SEVERE,
null, ex);
}
do{
    try{
        System.out.println("OK"); mess
        =input.readObject();
        System.out.println("not ok");
        if(mess instanceof String){
            if(!((String)mess).equals("CLOSE"))showMessage("\n" + mess);
        }
        else {
            ByteArrayInputStream bin=new ByteArrayInputStream((byte[]) mess);BufferedImage
            ms=ImageIO.read(bin);
            displayMessage(ms);
            bin.close();
        }
    }catch(Exception Ex){
        showMessage("\nThe server has sent an unknown object!");
    }
    System.out.println("INFINITE LOOP");
} while(mess instanceof byte[] || (mess!=null && !((String)mess).equals("CLOSE")));
}

public void closeConnection(){//////////sendMessage("CLOSE");
    showMessage("\nClosing Connections...");
    ableToSend(false);
    try{
        output.close();
        input.close();
        connection.close();
    }catch(IOException ioEx){
        showMessage("\nError in closing the streams!");
    }
}

private void sendMessage(Object message){//////////ByteArrayOutputStream bos;
    /*if(message==null){return;
    }*/
    try{
        if(message==null||message instanceof String){
            output.writeObject(message); output.flush();
        }else{BufferedImage bm=(BufferedImage)message;

```



```

        bos=new ByteArrayOutputStream();
        ImageIO.write(bm, "png", bos );
        byte[] imageInByte = bos.toByteArray();bos.flush();
        output.writeObject(imageInByte);
        output.flush(); bos.close();
    }
    setlabel.setText("No image is set!");Bufim=null;
} catch(IOException ioEx){
    statusarea.append("\n Error : Cannot send message!, PLEASE RETRY");
}
}

private void displayMessage(BufferedImage mes){ SwingUtilities.invokeLater(new
    Runnable(){
        @Override
        public void run() {
            if(mes==null){
                return;
            }
            Decode.setEnabled(true);
            BufferedImage IM=mes; JPanel
            Panel=new JPanel();
            Panel.setLayout(new FlowLayout(FlowLayout.LEFT));
            Panel.setBackground(Color.white);

            JCheckBox chk=new JCheckBox("SERVER");
            chk.setSize(100, 50); checks[++JC]=chk;
            grp.add(checks[JC]);

            JLabel lb=new JLabel();
            lb.setSize(150,150);
            lb.setIcon(new ImageIcon(IM.getScaledInstance(lb.getWidth(),
            lb.getHeight(),
            Image.SCALE_SMOOTH)));

            images[JC]=IM;

            Panel.add(lb); Panel.add(checks[JC]);
            chatbox.add(Panel);
            chatbox.add(Box.createVerticalStrut(20));cli.repaint();
            cli.validate();
        }
    });
}

private void showMessage(final String text){//////////////////////////////////SwingUtilities.invokeLater(
    new Runnable(){
        @Override
        public void run(){
            statusarea.append(text);
        }
    }
    );
}

private void ableToSend(final boolean tof){//////////////////////////////////SwingUtilities.invokeLater(

```

```

        new Runnable() {
            @Override
            public void run() {
                Send.setEnabled(tof);
            }
        });
    }
}
}

```

## EmbedMessage.java

```

package jdbcpkg;
//EmbedMessage.java  import
import java.awt.image.*;import
import javax.swing.*; import
import java.awt.*; import
import java.awt.event.*;import
import javax.imageio.*;

public class EmbedMessage extends JFrame implements ActionListener
{
    JButton open = new JButton("Open"), embed = new JButton("Embed"), set = new JButton("Done"),
    reset = new JButton("Reset");
    JTextArea message = new JTextArea(10,3);
    BufferedImage sourceImage = null, embeddedImage = null; JSplitPane sp = new JSplitPane(JSplitPane.HORIZONTAL_SPLIT);
    JScrollPane originalPane = new JScrollPane(),
    embeddedPane = new JScrollPane(); Serv SER=null;
    Clin CLI=null; String secur=null;

    public EmbedMessage(Object iim,String sec) {

        super("Embed stegonographic message in image"); secur=sec;
        if(iim instanceof Serv){
            SER=(Serv)iim;
        }
        else{
            CLI=(Clin)iim;
        }

        assembleInterface();

        this.setBounds(500,100,500, 500);
        this.setLocationRelativeTo(null);
        this.setDefaultCloseOperation(DISPOSE_ON_CLOSE);
        this.setVisible(true); sp.setDividerLocation(0.5);
        this.validate();
    }

    private void assembleInterface() {
        JPanel p = new JPanel(new FlowLayout());
        p.add(open); p.add(embed);
    }
}

```

```

p.add(set);
p.add(reset);
this.getContentPane().add(p, BorderLayout.SOUTH);
open.addActionListener(this); embed.addActionListener(this);
set.addActionListener(this); reset.addActionListener(this);
open.setMnemonic('O');
embed.setMnemonic('E'); set.setMnemonic('S');
reset.setMnemonic('R');

p = new JPanel(new GridLayout(1,1)); p.add(new
JScrollPane(message));
message.setFont(new Font("Arial",Font.BOLD,20)); p.setBorder(BorderFactory.createTitledBorder("Message to be
embedded")); this.getContentPane().add(p, BorderLayout.NORTH);

sp.setLeftComponent(originalPane);
sp.setRightComponent(embeddedPane);
originalPane.setBorder(BorderFactory.createTitledBorder("Original Image"));
embeddedPane.setBorder(BorderFactory.createTitledBorder("Steganographed Image"));
this.getContentPane().add(sp, BorderLayout.CENTER);
}

@Override
public void actionPerformed(ActionEvent ae) {Object o =
ae.getSource();
if(o == open)
    openImage();
else if(o == embed)
    embedMessage();
else if(o == set){
    setImage();
    this.dispose();
}
else if(o == reset)
    resetInterface();
}

private java.io.File showFileDialog(final boolean open) {JFileChooser
fc = new JFileChooser("Open an image");
javax.swing.filechooser.FileFilter ff = new javax.swing.filechooser.FileFilter() { @Override
public boolean accept(java.io.File f) { String name =
    f.getName().toLowerCase(); if(open)
        return f.isDirectory() || name.endsWith(".jpg") || name.endsWith(".jpeg") || name.endsWith(".png") || name.endsWith(".gif") ||
            name.endsWith(".tiff") || name.endsWith(".bmp") || name.endsWith(".dib");
        return f.isDirectory() || name.endsWith(".png") || name.endsWith(".bmp");
    }
}
@Override
public String getDescription() {
    if(open)
        return "Image (*.jpg, *.jpeg, *.png, *.gif, *.tiff, *.bmp, *.dib)";
    return "Image (*.png, *.bmp)";
}
};
fc.setAcceptAllFileFilterUsed(false); fc.addChoosableFileFilter(ff);
java.io.File f = null;

```

```

if(open && fc.showOpenDialog(this) == fc.APPROVE_OPTION)f =
    fc.getSelectedFile();
else if(!open && fc.showSaveDialog(this) == fc.APPROVE_OPTION)f =
    fc.getSelectedFile();
return f;
}

private void openImage() {
    java.io.File f = showFileDialog(true);
    try {
        sourceImage = ImageIO.read(f);
        JLabel l = new JLabel(new ImageIcon(sourceImage));
        originalPane.getViewport().add(l); this.validate();
    } catch(Exception ex) { ex.printStackTrace(); }
}

private void embedMessage() {
    if(sourceImage==null){
        JOptionPane.showMessageDialog(this, "Please choose an image !");
        return;
    }
    String mess = message.getText();embeddedImage =
    sourceImage; if(encode(embeddedImage, mess)){
        JLabel l = new JLabel(new ImageIcon(embeddedImage));
        embeddedPane.getViewport().add(l);
        this.validate();
    }
    else{
        embeddedImage=null;
    }
}

private void setImage() {
    if(embeddedImage == null) {
        JOptionPane.showMessageDialog(this, "No message has been embedded!", "Nothing to send",
        JOptionPane.ERROR_MESSAGE);
        return;
    }
    if(SER!=null){ SER.Bufim=embeddedImage;
        SER.setlabel.setText("Image is ready to be sent!");
    }
    else{
        CLI.Bufim=embeddedImage; CLI.setlabel.setText("Image is ready to
        be sent!");
    }
}

private void resetInterface() { message.setText("");
    originalPane.getViewport().removeAll();
    embeddedPane.getViewport().removeAll();
    sourceImage = null;
    embeddedImage = null; sp.setDividerLocation(0.5);
    this.validate();
}

    public boolean encode(BufferedImage img, String message)
    {
        return add_text(img,message);
    }

```

```

    }
    private boolean add_text(BufferedImage image, String text)
    {
        String s=secur;
        text+=s;
        char ln=(char)s.length();
        text+=ln;
        byte img[] = get_byte_data(image);
        byte msg[] = text.getBytes();
        byte len[] = bit_conversion(msg.length);
        try
        {
            encode_text(img, len, 0);
            encode_text(img, msg, 32);
        }
        catch(Exception e)
        {
            JOptionPane.showMessageDialog(null, "Target File cannot hold message!", "Error", JOptionPane.ERROR_MESSAGE);
            return false;
        }
        return true;
    }

    private byte[] get_byte_data(BufferedImage image)
    {
        WritableRaster raster = image.getRaster();
        DataBufferByte buffer = (DataBufferByte)raster.getDataBuffer();
        return buffer.getData();
    }

    private byte[] bit_conversion(int i)
    {
        byte byte3 = (byte)((i & 0xFF000000) >>> 24); byte byte2 =
        (byte)((i & 0x00FF0000) >>> 16); byte byte1 = (byte)((i &
        0x0000FF00) >>> 8 ); byte byte0 = (byte)((i & 0x000000FF)
        );
        return(new byte[] {byte3,byte2,byte1,byte0});
    }

    private byte[] encode_text(byte[] image, byte[] addition, int offset)
    {
        if(addition.length + offset > image.length)
        {
            throw new IllegalArgumentException("File not long enough!");
        }
        for(int i=0; i<addition.length; ++i)
        {
            int add = addition[i];
            for(int bit=7; bit>=0; --bit, ++offset)
            {
                int b = (add >>> bit) & 1;
                image[offset] = (byte)((image[offset] & 0xFE) | b );
            }
        }
        return image;
    }
}

```

## **DecodeMessage.java**

```
package jdbcpkg;
//DecodeMessage.java
import java.awt.image.*;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import javax.imageio.*;
public class DecodeMessage extends JFrame implements ActionListener
{
    JButton open = new JButton("Open"), decode = new JButton("Decode"), reset = new
    JButton("Reset");
    JTextArea message = new JTextArea(10,3);
    BufferedImage image = null;
    JScrollPane imagePane = new JScrollPane();
    public DecodeMessage(BufferedImage BM) { super("Decode
    steganographic message in image");image=BM;
    assembleInterface();

    this.setDefaultCloseOperation(DISPOSE_ON_CLOSE);
    this.setBounds(500,100,500,500); this.setVisible(true);
    }
    private void assembleInterface() {
        JPanel p = new JPanel(new FlowLayout());
        p.add(open);
        p.add(decode);
        p.add(reset);
        this.getContentPane().add(p, BorderLayout.NORTH);
        open.addActionListener(this); decode.addActionListener(this);
        decode.setEnabled(false); reset.addActionListener(this);
        open.setMnemonic('O');
        decode.setMnemonic('D');reset.setMnemonic('R');
        p = new JPanel(new GridLayout(1,1));p.add(new
        JScrollPane(message));
        message.setFont(new Font("Arial",Font.BOLD,20)); p.setBorder(BorderFactory.createTitledBorder("Decoded
        message"));message.setEditable(false);
        this.getContentPane().add(p, BorderLayout.SOUTH);

        imagePane.setBorder(BorderFactory.createTitledBorder("Steganographed Image"));
        this.getContentPane().add(imagePane, BorderLayout.CENTER);
    }
    @Override
    public void actionPerformed(ActionEvent ae) {Object o
    = ae.getSource();
    if(o == open)
        openImage(); else if(o == decode)
        decodeMessage();
    else if(o == reset) resetInterface();}
    private void openImage() {
    try { if(image==null){ JOptionPane.showMessageDialog(this, "No message has been selected !")
    return;}
        JLabel l = new JLabel(new ImageIcon(image));
        imagePane.getViewport().add(l);
        decode.setEnabled(true);this.validate();
    }
```

```

    } catch(Exception ex) { ex.printStackTrace(); }
}
private void decodeMessage() { message.setText(decode(image));
}
private void resetInterface() { message.setText("");
imagePane.getViewport().removeAll();image = null;
this.validate();
}
public String decode(BufferedImage img)
{
    class OTEEx extends Exception{String
        display(){
            return "Incorrect OTP!";
        }
    }
    byte[] decod;
    try
    {
        decod = decode_text(get_byte_data(img));String
        dc=new String(decod);
        String otp="";
        int Ch=(int)dc.charAt(dc.length()-1);
        for(int i=dc.length()-2;i>=dc.length()-1-Ch;i--){otp+=dc.charAt(i);
        }
        //otp recieved is reversed so i must reverse it again
        int ci=otp.length()-1;
        char[] ott=otp.toCharArray();
        for(int i=0;i<otp.length();i++){
            if(i<ci){
                char ch=ott[i];
                ott[i]=ott[ci];
                ott[ci]=ch;
                ci--;
            }
            else break;
        }
        dc=dc.substring(0, dc.length()-otp.length()-1);
        String oTp=JOptionPane.showInputDialog("Please enter the OTP!");
        boolean match=true;
        if(oTp.length()!=otp.length()){match=false;
        }
        if(match){
            for(int i=0;i<otp.length();i++){
                if(oTp.charAt(i)!=ott[i]){match=false;
                break;
            }
        }
        }
        if(!match){
            throw new OTEEx();
        }
        return(dc);
    }
    catch(OTEx ex){
        JOptionPane.showMessageDialog(null,
            ex.display(),"Error",

```

```

        JOptionPane.ERROR_MESSAGE);
        return "";
    }
    catch(Exception e)
    {
        JOptionPane.showMessageDialog(null, "Image
            not found!", "Error",
            JOptionPane.ERROR_MESSAGE);
        return "";
    }
}

private byte[] get_byte_data(BufferedImage image)
{
    WritableRaster raster = image.getRaster();
    DataBufferByte buffer = (DataBufferByte)raster.getDataBuffer();
    return buffer.getData();
}

private byte[] decode_text(byte[] image)
{
    int length = 0;
    int offset = 32;
    for(int i=0; i<32; ++i)
    {length = (length << 1) | (image[i] & 1);}
    byte[] result = new byte[length];for(int b=0;
    b<result.length; ++b )
    {for(int i=0; i<8; ++i, ++offset){result[b] = (byte)
        ((result[b] <<1) | (image[offset] &1))}return result;} }

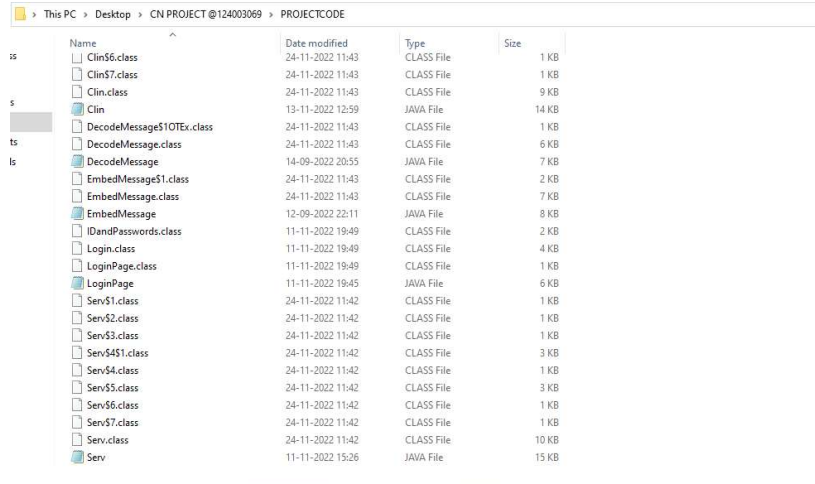
```

-----



## 4.13 IMPLEMENTATION OF THE PROJECT

Fig 11 Implementation snap shots



Name	Date modified	Type	Size
Clin\$6.class	24-11-2022 11:43	CLASS File	1 KB
Clin\$7.class	24-11-2022 11:43	CLASS File	1 KB
Clin.class	24-11-2022 11:43	CLASS File	9 KB
Clin	13-11-2022 12:59	JAVA File	14 KB
DecodeMessage\$1OTEx.class	24-11-2022 11:43	CLASS File	1 KB
DecodeMessage.class	24-11-2022 11:43	CLASS File	6 KB
DecodeMessage	14-09-2022 20:55	JAVA File	7 KB
EmbedMessage\$1.class	24-11-2022 11:43	CLASS File	2 KB
EmbedMessage.class	24-11-2022 11:43	CLASS File	7 KB
EmbedMessage	12-09-2022 22:11	JAVA File	8 KB
IDandPasswords.class	11-11-2022 19:49	CLASS File	2 KB
Login.class	11-11-2022 19:49	CLASS File	4 KB
LoginPage.class	11-11-2022 19:49	CLASS File	1 KB
LoginPage	11-11-2022 19:45	JAVA File	6 KB
Serv\$1.class	24-11-2022 11:42	CLASS File	1 KB
Serv\$2.class	24-11-2022 11:42	CLASS File	1 KB
Serv\$3.class	24-11-2022 11:42	CLASS File	1 KB
Serv\$4\$1.class	24-11-2022 11:42	CLASS File	3 KB
Serv\$4.class	24-11-2022 11:42	CLASS File	1 KB
Serv\$5.class	24-11-2022 11:42	CLASS File	3 KB
Serv\$6.class	24-11-2022 11:42	CLASS File	1 KB
Serv\$7.class	24-11-2022 11:42	CLASS File	1 KB
Serv.class	24-11-2022 11:42	CLASS File	10 KB
Serv	11-11-2022 15:26	JAVA File	15 KB

Command Prompt - java Clin

```
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

C:\Users\GOPINATH CHENNAMSETT>cd C:\Users\GOPINATH CHENNAMSETT\Desktop\CN PROJECT @124003069\PROJECTCODE

C:\Users\GOPINATH CHENNAMSETT\Desktop\CN PROJECT @124003069\PROJECTCODE>javac EmbedMessage.java

C:\Users\GOPINATH CHENNAMSETT\Desktop\CN PROJECT @124003069\PROJECTCODE>javac DecodeMessage.java

C:\Users\GOPINATH CHENNAMSETT\Desktop\CN PROJECT @124003069\PROJECTCODE>javac Clin.java

C:\Users\GOPINATH CHENNAMSETT\Desktop\CN PROJECT @124003069\PROJECTCODE>java Clin
```

Command Prompt - java Serv

```
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

C:\Users\GOPINATH CHENNAMSETT>cd C:\Users\GOPINATH CHENNAMSETT\Desktop\CN PROJECT @124003069\PROJECTCODE

C:\Users\GOPINATH CHENNAMSETT\Desktop\CN PROJECT @124003069\PROJECTCODE>javac Serv.java

C:\Users\GOPINATH CHENNAMSETT\Desktop\CN PROJECT @124003069\PROJECTCODE>javac EmbedMessage.java

C:\Users\GOPINATH CHENNAMSETT\Desktop\CN PROJECT @124003069\PROJECTCODE>javac DecodeMessage.java

C:\Users\GOPINATH CHENNAMSETT\Desktop\CN PROJECT @124003069\PROJECTCODE>java Serv
```

PROCEDURE TO EXECUTE : Open Two COMMAND Prompts and COMPILE EmbedMessage.java,DecodeMessage.java in Both prompts and compile Server in one and Client in another as shown in above

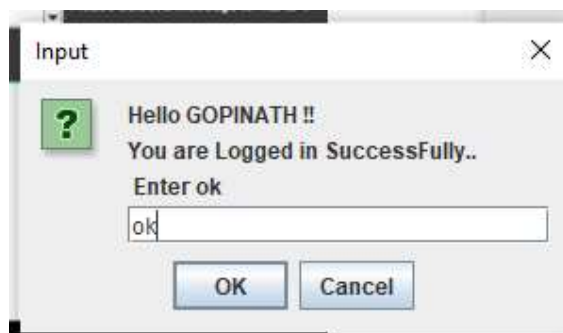
## CLIENT VERIFICATION AND REGISTRATION IN THE DATABASE

Fig 12 Output snap shots

CLIENT LOGIN FORM



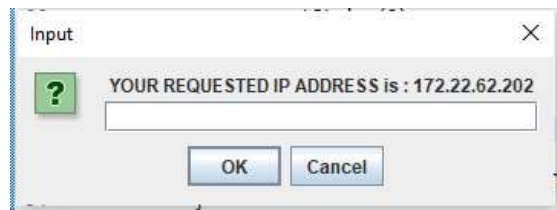
CLIENT NOT FOUND



CLIENT LOGGED IN



CLIENT REGISTRATION FORM



## INFORM IP ADDRESS TO LOGGED IN CLIENT TO FIND NETWORK INFORMATION

NETWORK INFORMATION

ENTER IP ADDRESS : 172.22.62.202

ENTER NUM OF ADDRESS IN SUBNET: 8

subnetdetails clear MAC

DISPLAYING MAC INFORMATION USING MAC ADDRESS RESOLUTION PROTOCOL  
IP ADDRESS : 172.22.62.202  
MAC ADDRESS : B0-52-16-A3-E2-E9

## MAC ADDRESS INFORMATION

NETWORK INFORMATION

ENTER IP ADDRESS : 172.22.62.202

ENTER NUM OF ADDRESS IN SUBNET: 8

subnetdetails clear MAC

DISPLAYING SUBNET INFORMATION  
Ip address: 172.22.62.202  
Binary Format 10101100000101100011111011001010  
Subnet mask = 29  
  
Network address is : 172.22.62.200.  
Broadcast address is = 172.22.62.207

## SUBNET INFORMATION

```
Run SQL Command Line
SQL*Plus: Release 11.2.0.2.0 Production on Mon Dec 5 23:36:44 2022
Copyright (c) 1982, 2014, Oracle. All rights reserved.

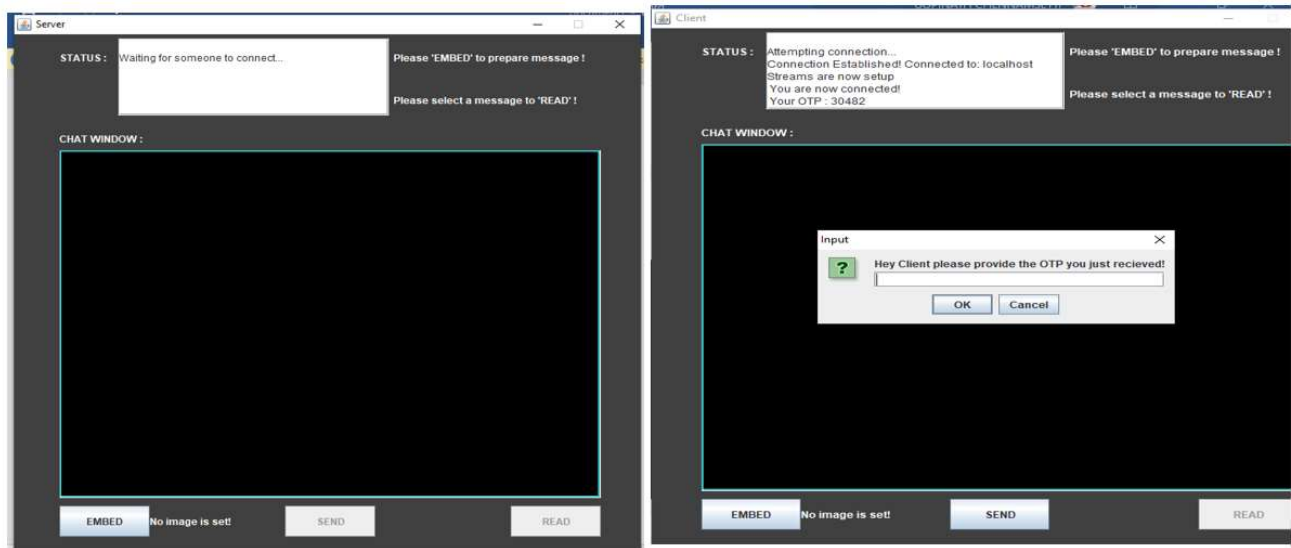
SQL> connect system
Enter password:
Connected.
SQL> select * from student;

REGNO      NAME      PASSWORD CGPA  BRANCH
-----
1234       qwer      qwerty   6.7   cse
12400400    RAVI      qwerfd   8.6   ece
124003069   GOPINATH  Gopi@123 9.0   CSE
111111111   maharshi  maha456  7.6   mec
124003590   ANIL      Anil@777 8.8   CSE
124003780   NARAYANA  Nara@123 8.5   CSE
124003901   SRINU     Svr@999  9.2   CSE
124003231   RAKESH    Raki@456 8.2   ECE
124003845   VASU      Vasu@666 9.1   EEE
jhgf       hgfg      jhgy     7.7   civil
22222222   ramu      ramu@345 8.5   cse

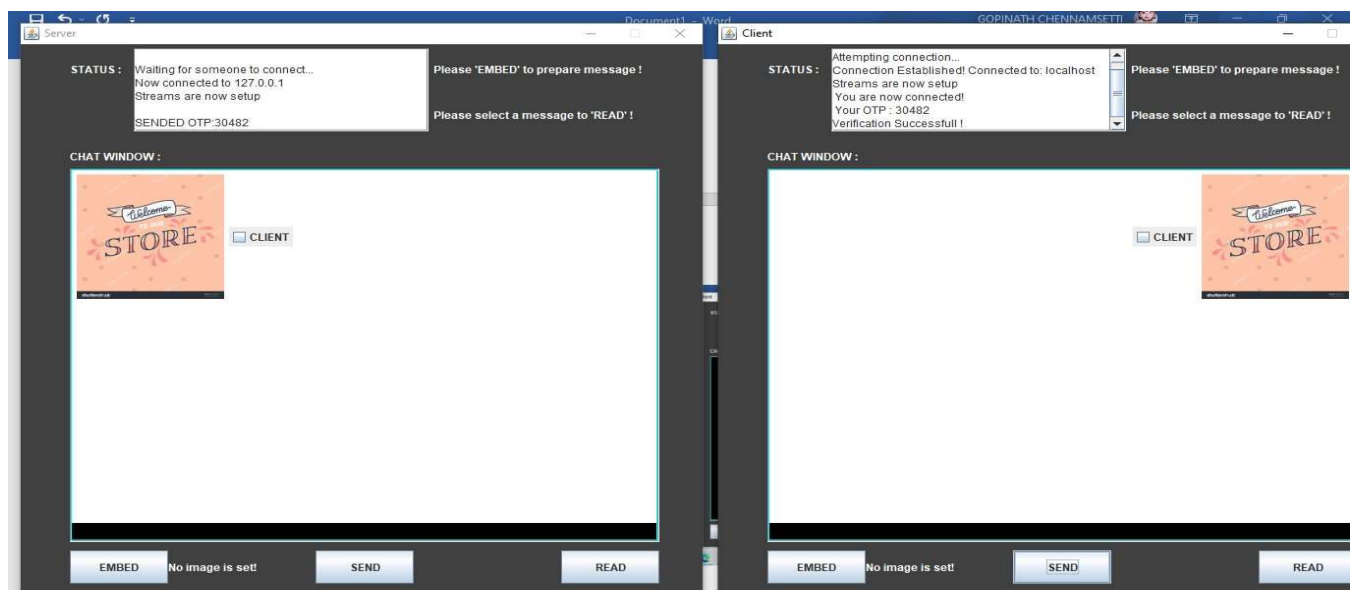
REGNO      NAME      PASSWORD CGPA  BRANCH
-----
123456     VASU      VASU@12  8.8   CSE

12 rows selected.
```

## DATA BASE OUTPUT

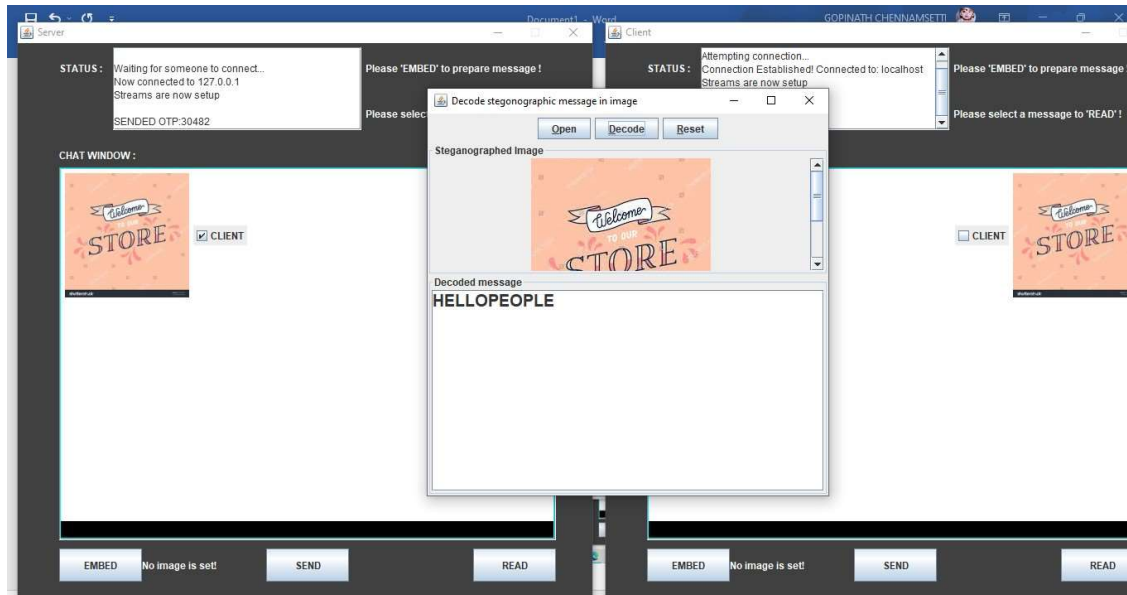


**SERVER WAIT FOR CONNECTIONS AND CLIENT REQUEST CONNECTION**

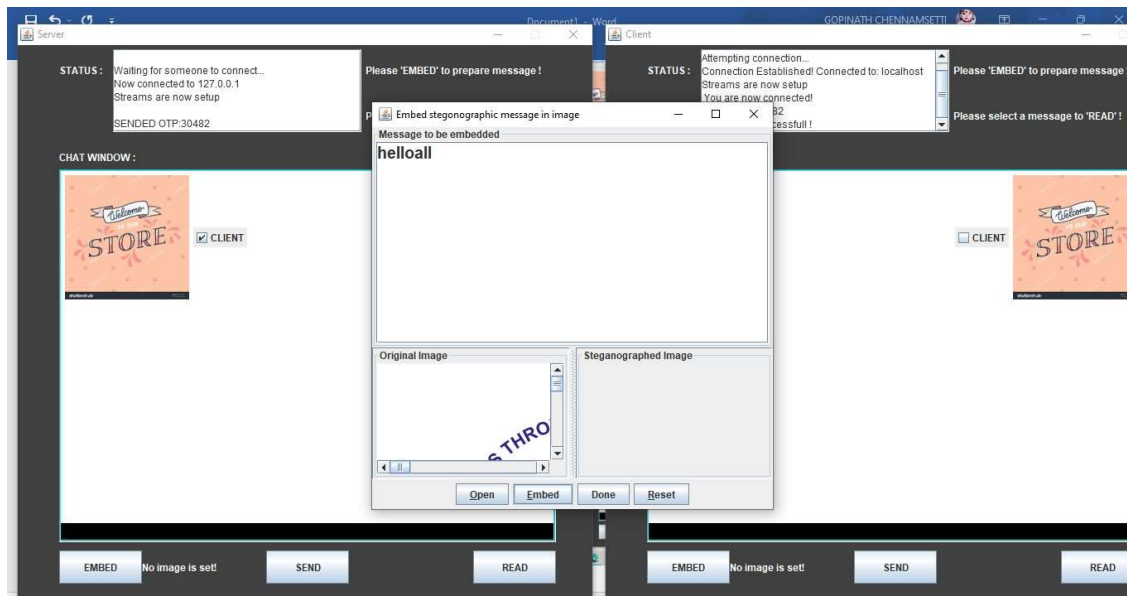


**CONNECTION ESATABLISHED AND ENCODED IMAGE SENT FROM CLIENT TO SERVER**

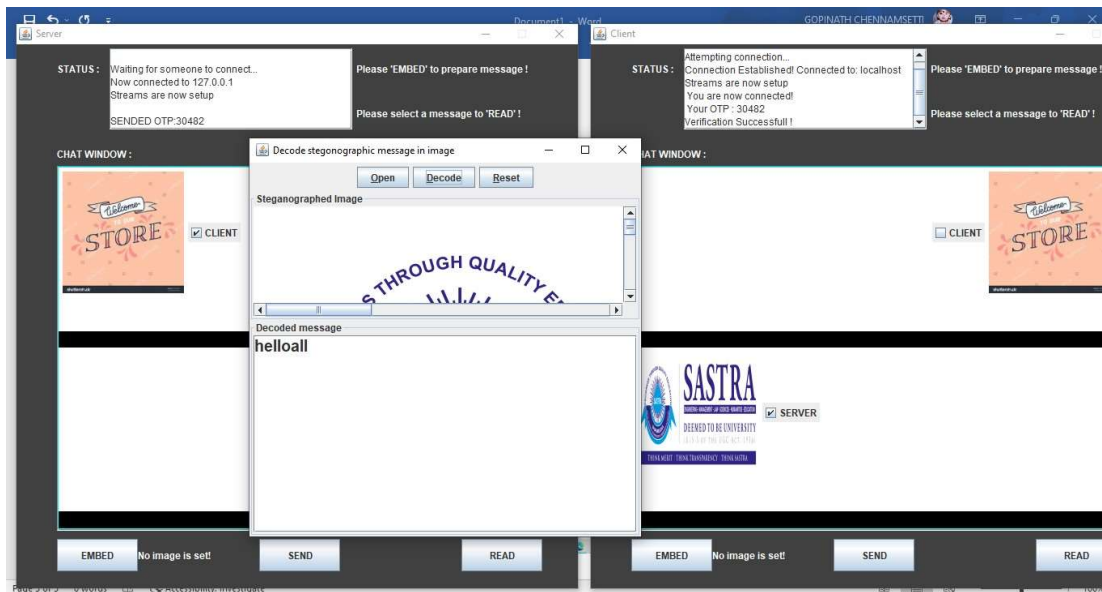
### 4.1.3 RESULTS AND SNAPSHOTS



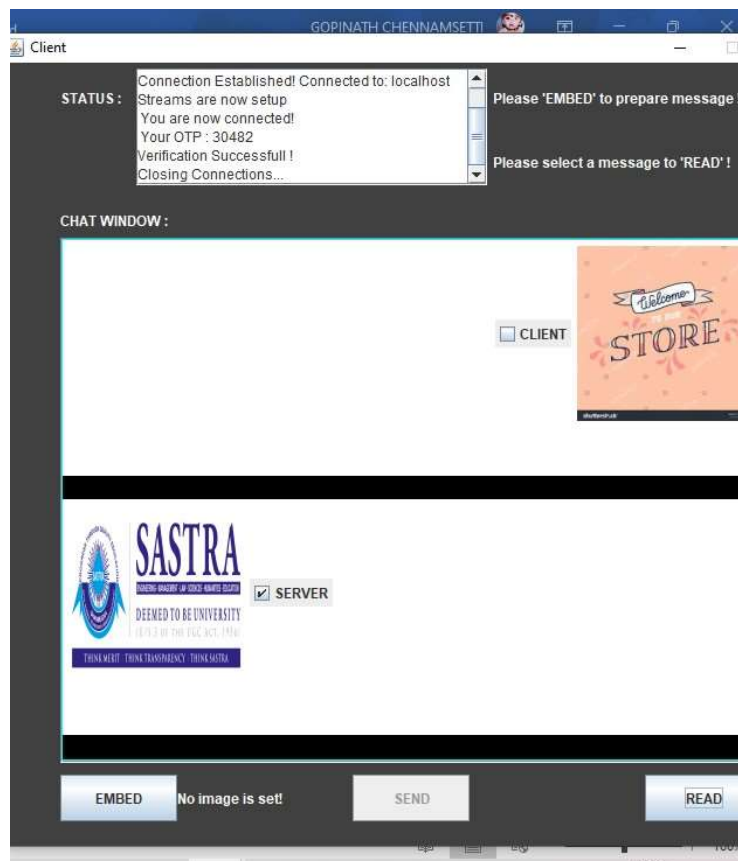
### DECODE MESSAGE AT SERVER



### ENCODE MESSAGE AT SERVER



## DECODE MESSAGE AT CLIENT



## CONNECTION CLOSED

## **CHAPTER FIVE**

### **CONCLUSION AND RECOMMANDATION**

#### **SUMMARY**

In summary, it is determined that the steganography system examined before is preferable. With the system flowchart and carefully chosen works consulted, the design is completed in sufficient depth to ensure success. By using a unique and fresh image when embedding data like audio, video, or images, we can reduce the likelihood that the attacker will find the data that is being concealed. The results obtained show that the security and robustness of our suggested strategy are promising. we demonstrated Network Information contains IP addresses manage the logical route-able connection from computer to computer and network to network while MAC addresses manage the actual connection between computers.

All of the software used to implement the new design system was run on a microcomputer. The software documentation, which is thoroughly examined in chapter four of this project, has an explanation of how the programme operates.

#### **CONCLUSION**

The information-delivery tool in this system is focused on making information accessible and secure. The data is key-encrypted and contains an image that is prepared for transmission across communication channels. It will be dependable and secure. The tool verifies the data's authenticity and verifies that it is available at the receiving end. It decrypts and gets data from the stego image. The personal information is contained inside the cover file image that the application creates as a stego image. This project developed the application using the Least Significant Bit algorithm, which is faster, more dependable, and has a moderate compression ratio when compared to other algorithms. There are two sessions in this package. Data embedding, retrieval, and authentication are the topics of the initial sessions. The tools for data encryption and decryption are provided in the second sessions.

## RECOMMENDATION

This project is a stand-alone, streamlined application that promotes data privacy. This approach can be utilised in a database record in the academic setting, where a variety of sensitive records are held (student outcome records, transcripts, accounts), making it difficult for unauthorised individuals to understand the records, much less change or edit them. The pledge of anonymity and non-repudiation is maintained, which lends the records some credence. This technology can also be used in the healthcare industry. In situations when a patient's medical report, record, and history need to be forwarded to a new hospital for additional treatment, or more, it offers a trustworthy channel because the patient cannot delete or change the encrypted history.

Additionally, since privacy and integrity improve their services and intrusion could cause a great deal of risk and damage, it is advised for corporate bodies (human resource enterprises) that have a lot of sensitive information and risk to manage to encrypt and safeguard messages and information available to them. Lastly, it should be added as one of the many ways for IT security companies to guarantee more security for information.

## REFERENCES :

- <http://en.wikipedia.org/wiki/steganography>
- . Amitava Nag , Sushanta Biswas , Debasree Sarkar ,ParthaPratim Sarkar. (n.d.). A Novel Technique for Image Steganography Based on DWT and Huffman Encoding. International Journal of Computer Science and Security, 4(6).
- 2. Elham Ghasemi, JamshidShanbehzadeh, NimaFassih. (2011, March). High Capacity Image Steganography Using Wavelet Transform and Genetic Algorithm. proceedings of the international multiconference of ngineers and ComputerScientistsi, ”, IMECS Hon, 1(1), 16-18.
- <https://www.javatpoint.com/socket-programming>
- <https://www.geeksforgeeks.org/how-address-resolution-protocol-arp-works/>
- <https://www.techtarget.com/searchnetworking/tip/IP-addressing-and-subnetting-Calculate-a-subnet-mask-using-the-hosts-formula>
- <https://www.javatpoint.com/java-jdbc>



