Data Security using Cryptography and Steganography

1. Abstract and keywords

- Data Security using cryptography and steganography is yet another learning time project developed in android, the idea about it came when I was studying Information Security in my V SEM. And then and there I thought I should come up with something related to this.
- So that's how this cryptography came into reality for me. Till now it is available for desktop, I will be creating an android app so that it will help many people to keep their data safe. What it does is, it simply takes an input file and then converts it into some non-readable (cipher text) format, with some password, and after receiving it,it will decrypt using the same password.
- So in this way the sender will send the Non-Readable text to the receiver using the public network and without taking any risk, as in the meantime if some unauthorized user will try to read it then he will not make out any sense out of what non-readable cipher text contains. So at the receiver's end upon receiving the cipher text message he will use this Encryption System and the same password used by the sender, to decrypt it into the original format.
- So here, one more concept we are going to use i.e data hiding (Steganography). By using steganography data (secret information) can hide within data (multimedia data, here multimedia data is an image) and it can be sent anywhere to transfer the message easily without giving any suspicion to others.
- The difference lies in the fact that steganography hides the data so that nothing appears out of ordinary while cryptography encrypts the text, making it difficult for an outsider to infer anything from it even if they do attain the encrypted text. Both of them are combined to increase the security against various malicious attacks.

2. Introduction

- Cryptography is also referred to as data encryption while steganography is also called data hiding. Due to faster growth in multimedia technology, internet and cell phones, there is a need for security. Therefore there is a need for encryption techniques in order to hide the data from such attacks.
- Today the prominence of the internet day to day increased a lot and the transfers of files and confidential information over the internet demands the security and safety of the files and this can be accomplished by using encryption and decryption. In the current scenario, encryption and decryption are most widely used in every field like banking, etc.
- This project has a similar mechanism to provide the security and safety of the files by using a symmetric key. Also, hiding the message using the concept of steganography. The Files are only encrypted and decrypted by using the same key so that no one can access it.

2.1 Problem Statement

The problem of Data Security using cryptography and steganography can be stated as followed:

Hiding data on mobile app, end to end communication.

2.2 Literature Review/Description of Present System

- Current system is only available for desktop in which we can encrypt the data using the key and will store in some folder, while decrypting we will use the same key.
- It is done using the AES algorithm.
- In Current System, we can hide also data in the images.
- It is only available for the desktop.

2.3 Background /Limitations

- It is not strongly encrypted.
- There is no platform for sending and receiving hidden data.
- Only available for desktop.
- Mobile users cannot use application as the software is desktop based application

2.4. Aim & Objectives

- This project is proposed for the betterment of society.
- This project aims to help the people for sharing the data securely without any fear.
- The success of this project will also encourage developers to build something more useful for people, who also deserve an equal standard in society.

2.5. Project Motivation

- The Project Motivation is, according to a survey, around 71.1 million people fall victim to cybercrimes yearly.
- That means around 71.1 million people are deprived or unaware of the Cryptography and Steganography.
- They are fully depend on the online security. But there are less apps available for the data security.
- So, this app is designed and engineered especially for people who want to send their data securely without any fear.

03. Description of Proposed Work

- Data Security using cryptography and steganography is an android app which may help many people with the challenge of the security.
- In this app, we can encrypt the data and can send that encrypted data to the person whoever we want to send.
- In between, no one is going to access the data.
- After receiving the encrypted text, the receiver will decrypt it using the same app.
- Also, the user can encrypt the images too.
- There is one more concept that is steganography, in which user can hide their data within the images and this image will get stored into the Download folder. After using the same app,the user can decrypt the encrypted image.

3.1 Number of Modules

1. Login:-

The Login Module is module that allows users to type a user name and password to log in.

2. Home Module:-

It will display the Home page of the android app.

3. Crypto module for Text:

For Crypto Module the following steps are considered for encrypting the data

- Insert text for encryption.
- Apply AES algorithm using 128 bit key (Key 1).
- Generate Cipher Text in hexadecimal form

4. Crypto Module (Reverse Process)for Text:

For Crypto Module the following steps are considered for retrieving the original text.

- Get the above retrieved cipher text.
- Reverse AES algorithm by using Key 1.
- Get the original message.

5. Crypto module for Images:

In this module, first we will select image from the folder and then the image will get encrypted (Converting it into non-readable text)

6. Crypto Module(Reverse Process)for Images:

In this module, first we will enter the cipher text into the textfield and through that we will get the actual image out of it.

7. Stego Module:

Stego Module include the following steps:

- Choose the Image
- Enter the message and key
- Click on encode (Hiding the message into the image)
- Then the Encoded message will get printed.
- After encoding, we can save that encoded images into the Download folder.

8. Stego Module (Reverse Process) for Retrieving Hidden Text:

- Choose the Image(Encoded Image)
- Then enter the Secret Key
- Click on Decode
- Then the original text will get displayed into the textbox.
-

9. Security Module:

- Here, after encrypting the text ,the sender can send that encrypted message to other person using their phone number.(The app should be present on both the mobile)
- The receiver will receive it and decrypt it using the cryptable app.
- It allows you to send and receive the hidden messages.

3.2 Algorithm

This project uses the AES algorithm for Cryptography and Least Significant Bit for Steganography.

- **AES(Advanced Encryption Standard):**

```
setKey(secret);
            Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
            cipher.init(Cipher.ENCRYPT_MODE, secretKey);
            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
                return
Base64.getEncoder().encodeToString(cipher.doFinal(strToEncrypt.getBytes("UTF-8")));
```

- AES is Symmetric key algorithm and was Proposed by Rijndal and isalso known as the Rijndal encryption algorithm.
- It is a replacement of DES(Data Encryption Standard)
  AES algorithm works on block cipher technique and the size of the plain text must be same as cipher text.
- An Input Key is also required input to AES algorithm i.e same size of plain text.
  In AES,the data length(the plain text size) of 128,192 and 256 bits, and supporting three different key lengths,128,192 and the 256 bits.
  AES consist of multiple rounds:
  10 rounds →128-bit keys
  12 rounds →192-bit keys
  14 rounds →256-bit keys

❖ Steps for AES Algorithm:

- Plain Text transform in Matrix Form
  Example: “AES USES A MATRIX”
  Plain text converts into 4*4 matrix of bytes
  Text

A	E	S	U	S	E	S	A	M	A	T	R	I	X	Z	Z

Hexadecimal

00	04	12	14	12	04	12	00	0C	00	13	11	08	17	19	19

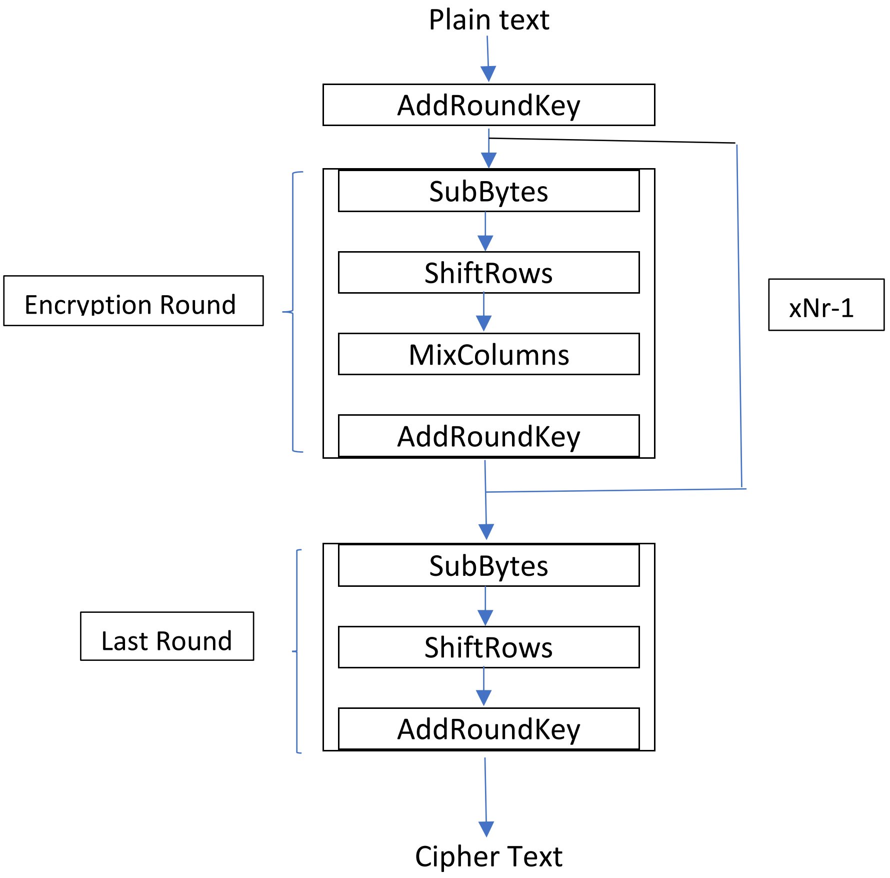
Therefore,the first four bytes of 128-bit input block occupy first column in 4x4 matrix of bytes.The next four bytes occupy the second column and so on.

It operates on 4*4 column-major order matrix of the bytes called state array.

00	12	0C	08
04	04	00	17
12	12	13	19
14	00	11	19

[state]

Overall structure of AES encryption process:



For encryption, each round consist of the following steps:-

1) **SubBytes**

Each individual byte of state is mapped into a new byte in a following way:The leftmost 4 bytes are used as a row value and the rightmost 4 bits are used as a column value. These row and columns serves as indexes into th S-box to select a unique 8-bit output value.

2) **ShiftRows**

The shift row transformation is called ShiftRows.
Rules:
Row 1: No Shifting
Row 2: 1 byte left Shifting
Row 3: 2 byte left Shifting
Row 4: 3 byte left Shifting

3) **MixColumns**

The mix Column transformation, called MixColumns ,operates on each column individually.

Each byte of a column is mapped into a new value that is a function of all four bytes in that column.

4) **AddRoundKey**

In this, 128 bits of state are bitwise XORed with the 128 bits of round key.

These are the steps for the AES Algorithm. The decryption is simply the inverse of the encryption.

❖ Comparison between AES and DES.

- In AES, Plain Text can be 128,192 or 256 bits.
- In DES, Plain Text is of the 64 bits.
- AES has the larger Key size as compared to DES.
- DES in comparison to AES has smaller key size.
- AES has a large secret key comparatively hence,more secure.DES has a smaller key which is a less secure.
- AES is faster. DES is comparatively slower.

❖ Least Significant Bit:

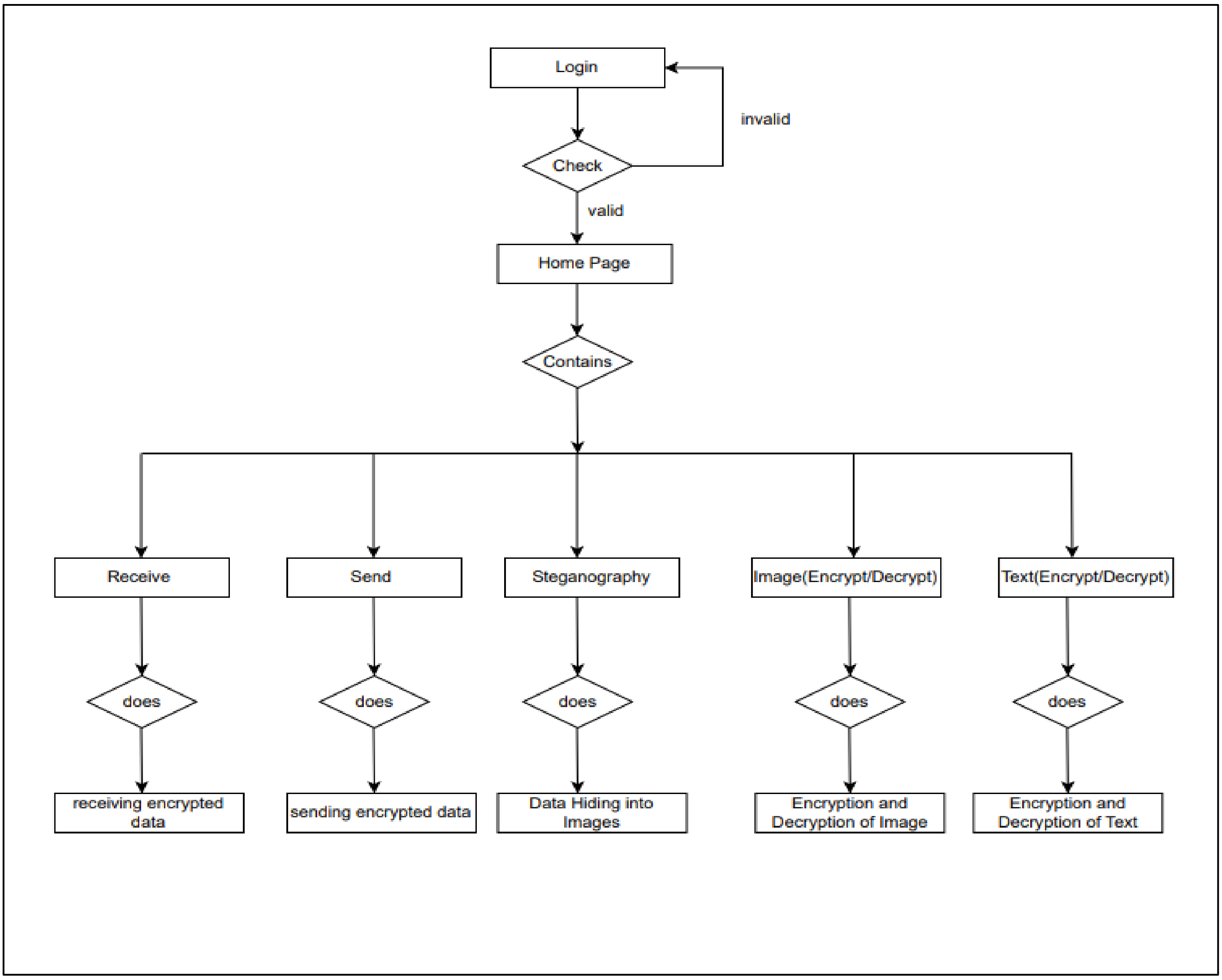
- It is lowest significant byte value of Image pixel and the Least Significant bit based image steganography embeds secret in the least significant bit of pixel value of CVR.
- The concept of the Least Significant Bit Embedding is simple that it exploits the fact that level of the precision in many image formats is far greater than that perceivable human vision.
- Therefore, an altered image with the slight variations in colors will be indistinguishable from original by the humans.
- In Least Significant Bit, just four bytes of the pixels are the sufficient to hold one of the message byte.Rest bits in pixels remains the same.

3.3 Working

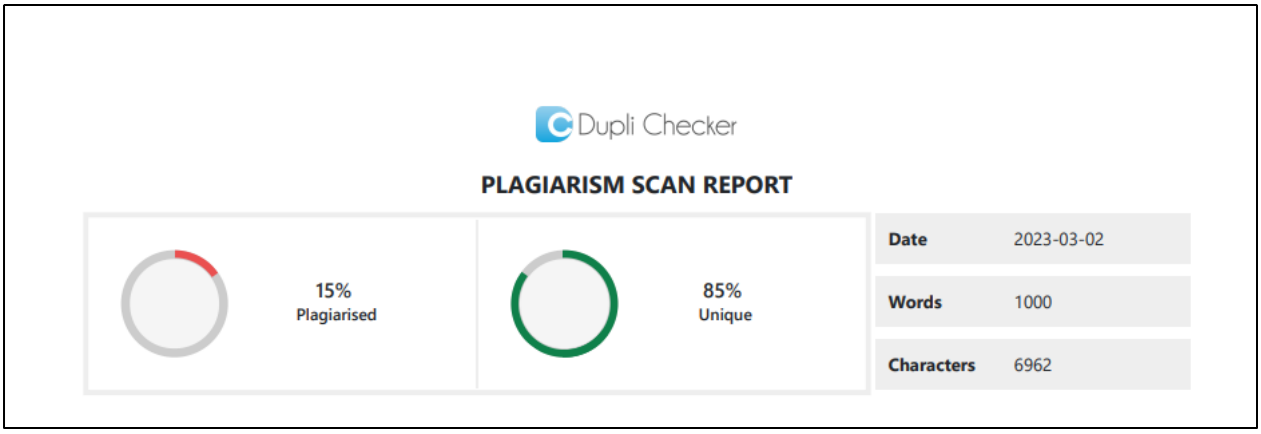
- This project will be in the form of an Android App.
- The project described within this document allows the people to send and receive the data securely.
- First there is a login page, so that user can successfully login using the username and password.
- After that there will be options like Text Encrypt/Decrypt,Image Encrypt/Decrypt and the Steganography.

- In Text (Encrypt/Decrypt),we can encrypt or decrypt the text using the AES algorithm.
- Same for the Images(Encrypt/Decrypt),we can encrypt or decrypt the images.
- There is option i.e Steganography in which user can hide the data in images using the key and message, after that the image will get saved into the download folder.At the time of Decryption, the user can use the same key for decrypting the data from the images.
- There is one more option of sending and receiving of data,in which the user needs to enter the data and number(mobile number) of person whom they want to send the message. After sending the data, the receiver will receive it in the same App.

3.4 Design/Block diagram/ flow chart/ graph/ deployment diagram/Architectural Design



3.5 Plagiarism report



Dupli Checker

PLAGIARISM SCAN REPORT

15%
Plagiarised

85%
Unique

Date	2023-03-02
Words	1000
Characters	6962

3.6 Coding:

- **File Name: activity_text_encoder.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".TextEncoder"
    android:background="@drawable/background">

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <androidx.constraintlayout.widget.ConstraintLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <ImageView
                android:id="@+id/imageView4"
                android:layout_width="90dp"
                android:layout_height="90dp"
                android:layout_marginTop="60dp"
                android:elevation="7dp"
                android:src="@drawable/cryptable"
                app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginHorizontal="30dp"
        android:layout_marginVertical="105dp"
        android:layout_marginBottom="30dp"
        android:background="@color/white"
        android:elevation="5dp"
        android:orientation="vertical"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginHorizontal="15dp"
            android:layout_marginTop="60dp"
            android:orientation="horizontal">

            <ImageView
                android:layout_width="40dp"
                android:layout_height="40dp"
                android:src="@drawable/enter_text" />

            <EditText
                android:id="@+id/enter_text_editText"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginLeft="5dp"
                android:textSize="14sp"
                android:backgroundTint="#00000000"
                android:hint="Enter you text here..."
                android:textColor="@color/navy_blue"
                android:textColorHint="@color/navy_blue" />
        </LinearLayout>

        <Button
            android:id="@+id/encrypt_btn"
            android:layout_width="match_parent"
            android:layout_height="60dp"
            android:layout_marginHorizontal="30dp"
            android:layout_marginTop="20dp"
            android:backgroundTint="@color/pastel"
            android:text="Encrypt"
            android:textColor="@color/white"
    android:textSize="18sp"
    android:onClick="enc"/>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginHorizontal="15dp"
    android:layout_marginTop="30dp"
    android:orientation="horizontal">

    <ImageView
        android:layout_width="40dp"
        android:layout_height="40dp"
        android:src="@drawable/enter_code" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginLeft="5dp"
        android:backgroundTint="#00000000"
        android:text="Your encrypted text:-"
        android:textColor="@color/navy_blue"
        android:textColorHint="@color/navy_blue"
        android:textSize="12sp" />
</LinearLayout>

<TextView
    android:id="@+id/tv_encrypted_txt"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:layout_marginHorizontal="15dp"
    android:layout_marginTop="20dp"
    android:backgroundTint="#00000000"
    android:text=""
    android:textColor="@color/navy_blue"
    android:textColorHint="@color/navy_blue"
    android:textSize="14sp" />
<Button
    android:id="@+id/copy_text_btn"
    android:layout_width="match_parent"
    android:layout_height="60dp"
    android:layout_marginHorizontal="30dp"
    android:layout_marginTop="20dp"
    android:layout_marginBottom="40dp"
    android:textColor="@color/white"
    android:backgroundTint="@color/pastel"
    android:text="Copy Text"
```

```
                android:textSize="18sp"
                android:onClick="copy"/>
            </LinearLayout>
        </androidx.constraintlayout.widget.ConstraintLayout>
    </ScrollView>
</androidx.constraintlayout.widget.ConstraintLayout>
```

- **File Name: TextEncoder.java**

```
package com.example.cryptable;
import androidx.appcompat.app.ActionBar;
import androidx.appcompat.app.AppCompatActivity;
import android.content.ClipData;
import android.content.ClipboardManager;
import android.content.Context;
import android.os.Bundle;
import android.view.View;
import android.view.inputmethod.InputMethodManager;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class TextEncoder extends AppCompatActivity {

    EditText enterText;
    TextView encryptedText;
    Button encryptBtn, copyTextBtn;
    ClipboardManager clipboardManager;
    final String secretKey = "akshadaa";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_text_encoder);

        ActionBar actionBar = getSupportActionBar();
        actionBar.setHomeButtonEnabled(true);
        actionBar.setTitle("Text Encryption");
        actionBar.show();

        enterText = findViewById(R.id.enter_text_editText);
        encryptedText = findViewById(R.id.tv_encrypted_txt);
        encryptBtn = findViewById(R.id.encrypt_btn);
        copyTextBtn = findViewById(R.id.copy_text_btn);
        clipboardManager = (ClipboardManager)
getSystemService(Context.CLIPBOARD_SERVICE);
```

```
    }
    public void enc(View view){
        String temp = enterText.getText().toString();
        String rv = AESText.encrypt(temp, secretKey);
        encryptedText.setText(rv);
        closeKeyboard();
    }
    public void copy(View view){
        String data = encryptedText.getText().toString().trim();
        if(!data.isEmpty()){
            ClipData temp = ClipData.newPlainText("text", data);
            clipboardManager.setPrimaryClip(temp);
            Toast.makeText(this, "Copied to clipboard!", Toast.LENGTH_SHORT).show();
        }
    }
    private void closeKeyboard() {
        View view = this.getCurrentFocus();
        if (view != null) {
            InputMethodManager imm = (InputMethodManager)
getSystemService(Context.INPUT_METHOD_SERVICE);
            imm.hideSoftInputFromWindow(view.getWindowToken(), 0);
        }
    }
}
```

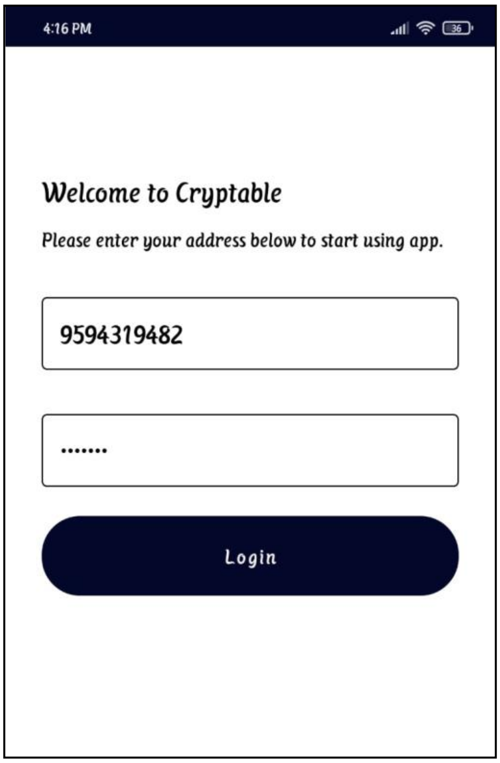
- **File Name: AESText.java**

```
package com.example.cryptable;
import android.os.Build;
import java.io.UnsupportedEncodingException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.Arrays;
import java.util.Base64;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;

public class AESText {

    private static SecretKeySpec secretKey;
    private static byte[] key;

    public static void setKey(String myKey)
    {
        MessageDigest sha = null;
        try {
            key = myKey.getBytes("UTF-8");
```

```
         sha = MessageDigest.getInstance("SHA-1");
         key = sha.digest(key);
         key = Arrays.copyOf(key, 16);
         secretKey = new SecretKeySpec(key, "AES");
      }
      catch (NoSuchAlgorithmException e) {
         e.printStackTrace();
      }
      catch (UnsupportedEncodingException e) {
         e.printStackTrace();
      }
   }
   public static String encrypt(String strToEncrypt, String secret) {
      try
      {
         setKey(secret);
         Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
         cipher.init(Cipher.ENCRYPT_MODE, secretKey);
         if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            return
Base64.getEncoder().encodeToString(cipher.doFinal(strToEncrypt.getBytes("UTF-8")));
         }
      }
      catch (Exception e)
      {
         System.out.println("Error while encrypting: " + e.toString());
      }
      return null;
   }

   public static String decrypt(String strToDecrypt, String secret)
   {
      try
      {
         setKey(secret);
         Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5PADDING");
         cipher.init(Cipher.DECRYPT_MODE, secretKey);
         if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            return new String(cipher.doFinal(Base64.getDecoder().decode(strToDecrypt)));
         }
      }
      catch (Exception e) {
         System.out.println("Error while decrypting: " + e.toString());
      }
      return null;
   }
}
```

3.7 Screen Layouts

Login

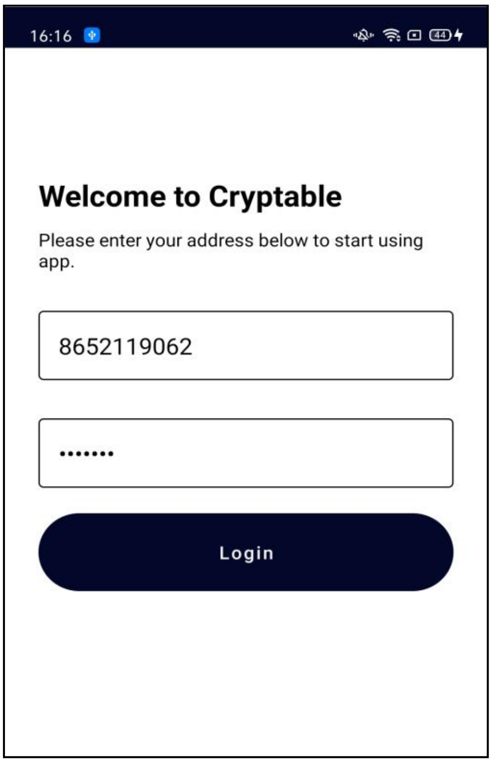


Welcome to Cryptable

Please enter your address below to start using app.

9594319482

•••••••



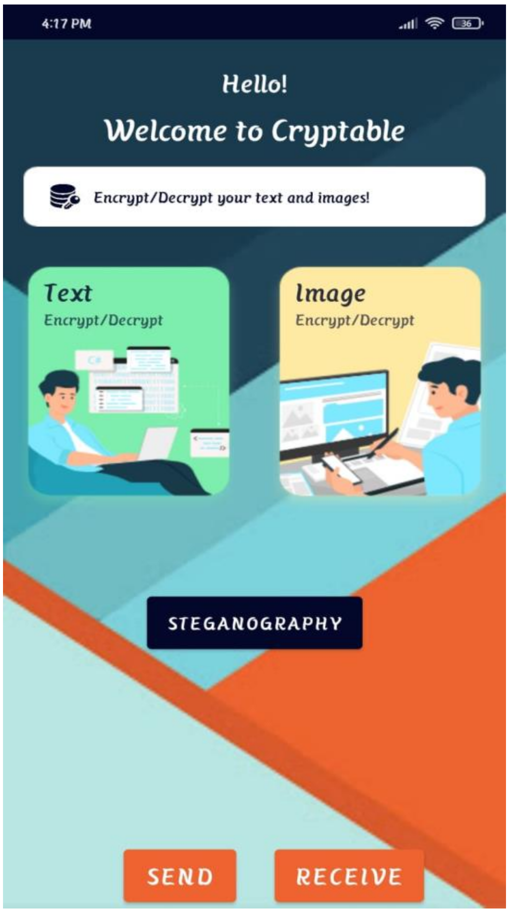
Welcome to Cryptable

Please enter your address below to start using app.

8652119062

•••••••

HomePage

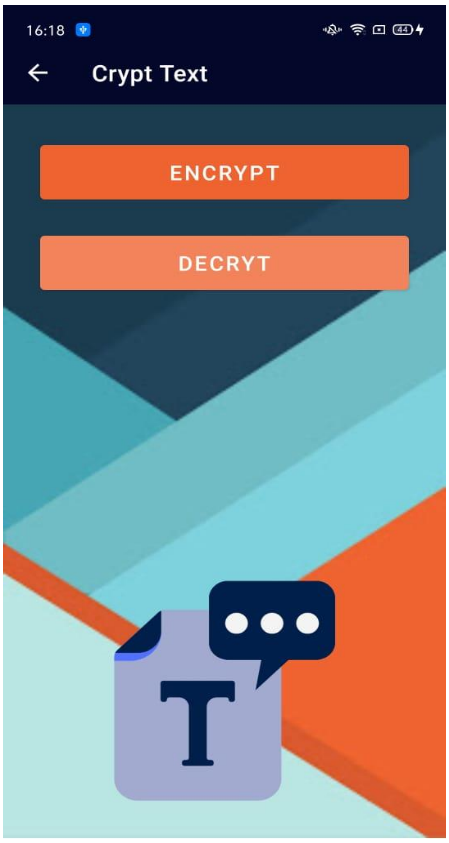


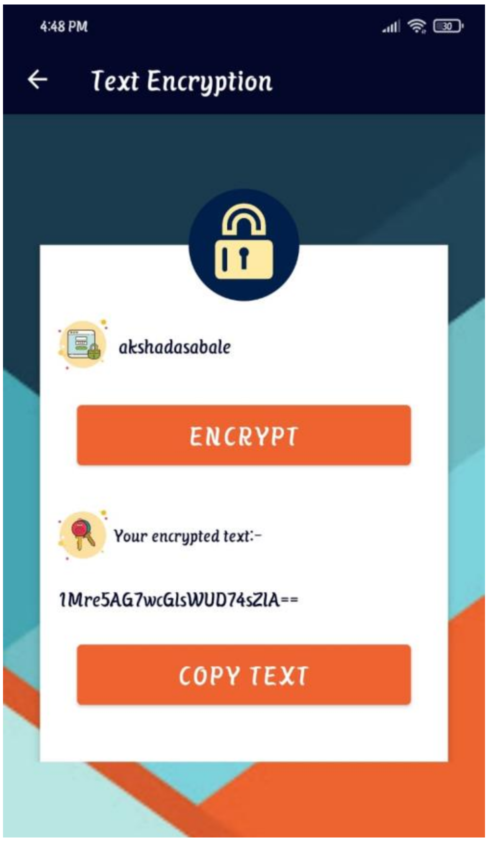
Hello!

Welcome to Cryptable

Encrypt/Decrypt your text and images!

Encryption of Text



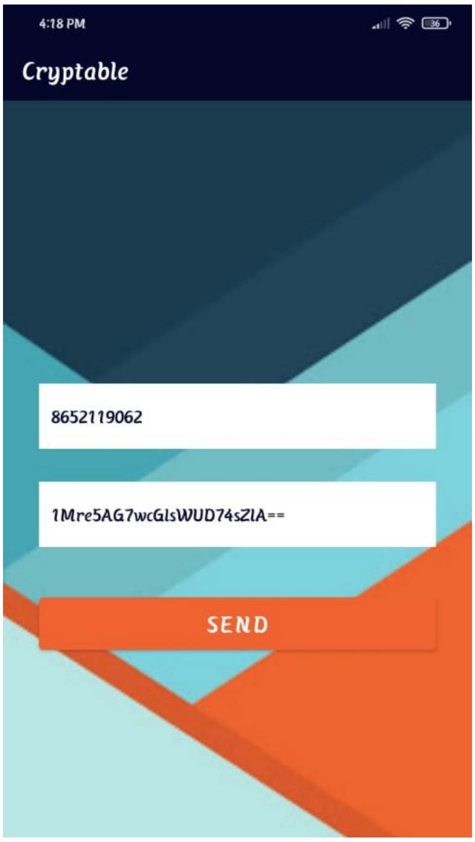


akshadasabale

Your encrypted text:-

1Mre5AG7wcGlsWUD74sZlA==

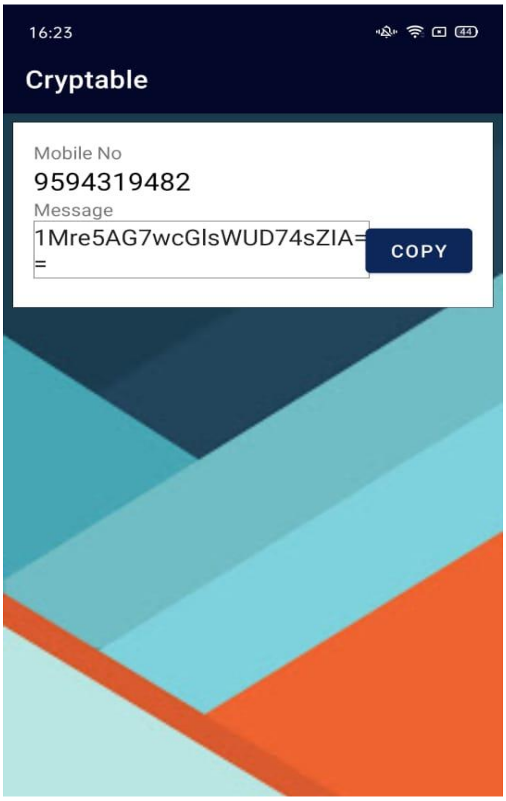
Sending Encrypted Text



8652119062

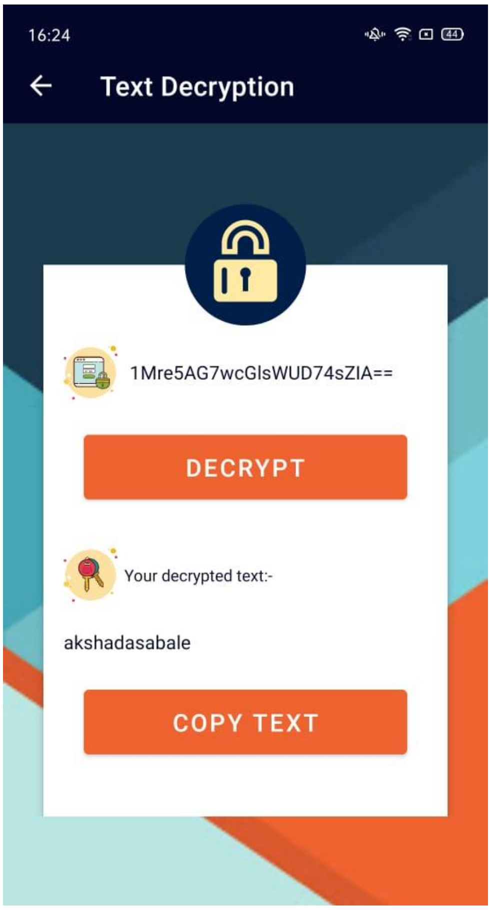
1Mre5AG7wcGlsWUD74sZlA==

Receiving Encrypted Text



Mobile No
9594319482
Message
1Mre5AG7wcGlsWUD74sZIA==

Decryption of Text

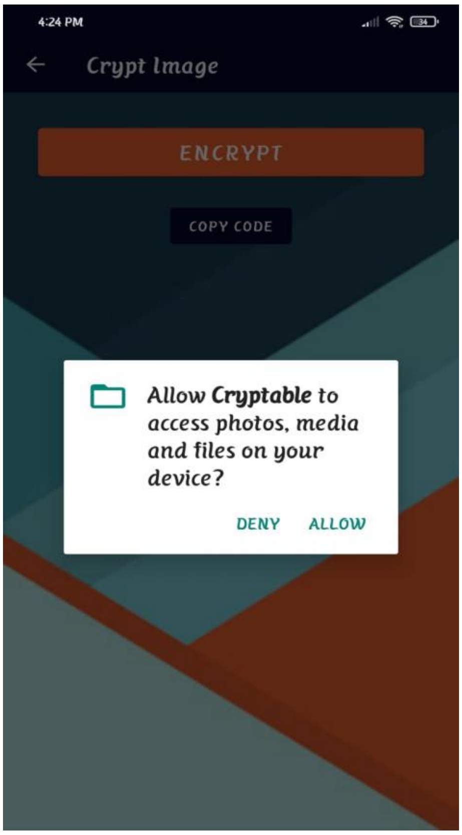


1Mre5AG7wcGlsWUD74sZIA==

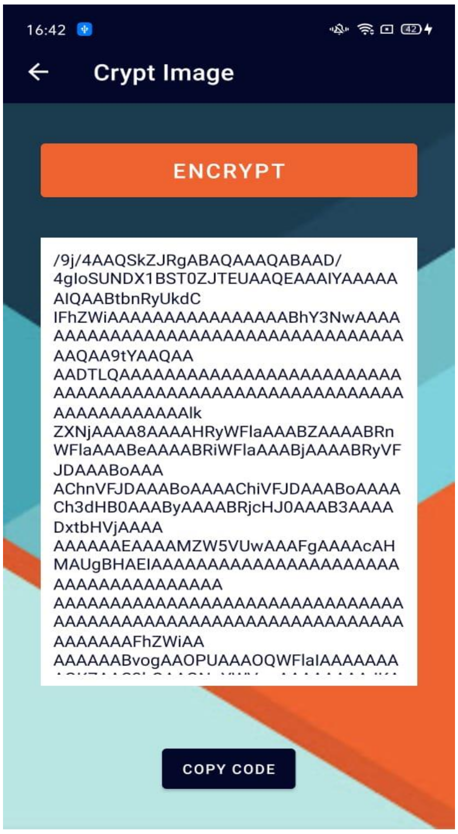
Your decrypted text:-

akshadasabale

Encryption of Image

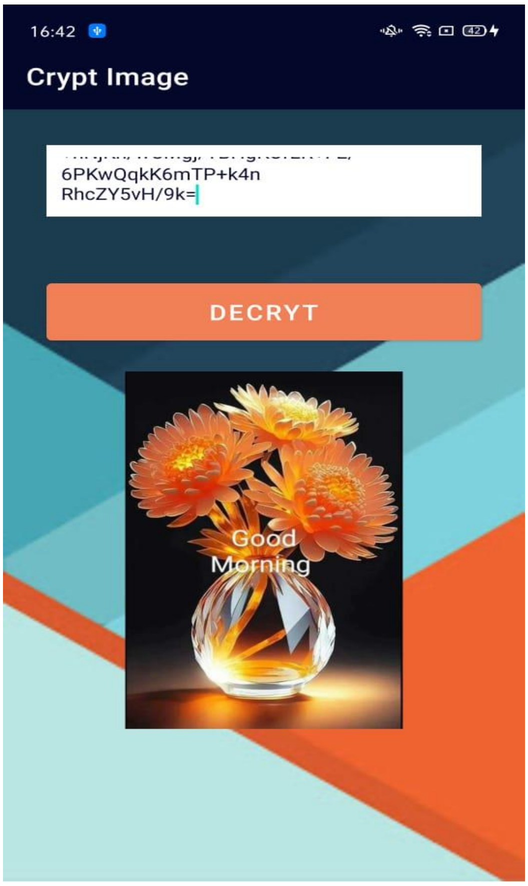


Allow **Cryptable** to access photos, media and files on your device?



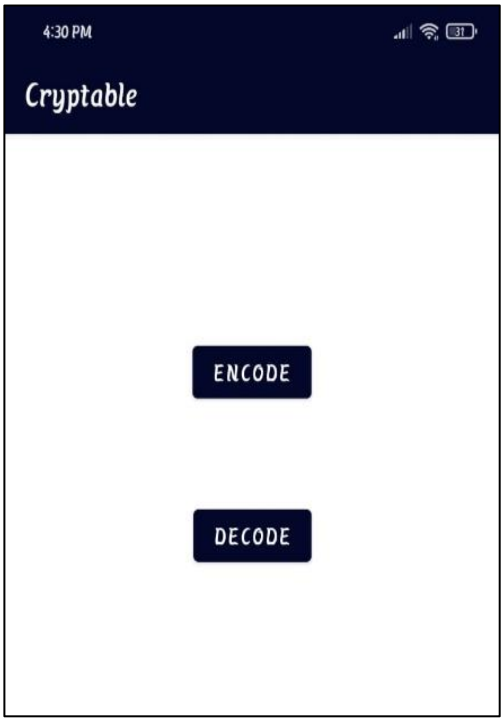
```
/9j/4AAQSkZJRgABAQAAAQABAAD/
4gIoSUNDX1BST0ZJTEUAAQEAAAIYAAAAA
AIQAABtbnRyUkdC
IFhZWiAAAAAAAAAAAAAAAAAABhY3NwAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAQAA9tYAAQAA
AADTLQAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAIk
ZXNjAAAA8AAAAHRyWFlaAAABZAAAABRn
WFlaAAABeAAAABRiWFlaAAABjAAAABRyVF
JDAAABoAAA
AChnVFJDAAABoAAAAChiVFJDAAABoAAAA
Ch3dHB0AAAByAAAABRjcHJ0AAAB3AAAA
DxtbHVjAAAA
AAAAAAEAAAAMZW5VUwAAAFgAAAAcAH
MAUgBHAEIAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAFhZWiAA
AAAAAABvogAAOPUAAAOQWFlaIAAAAAAA
```

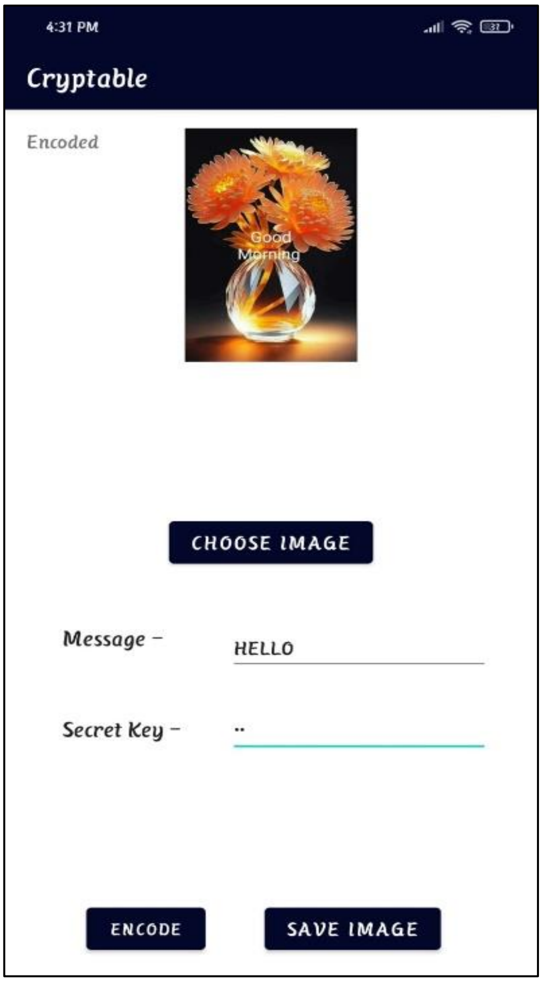
Decryption of Image



```
6PKwQqkK6mTP+k4n
RhcZY5vH/9k=
```

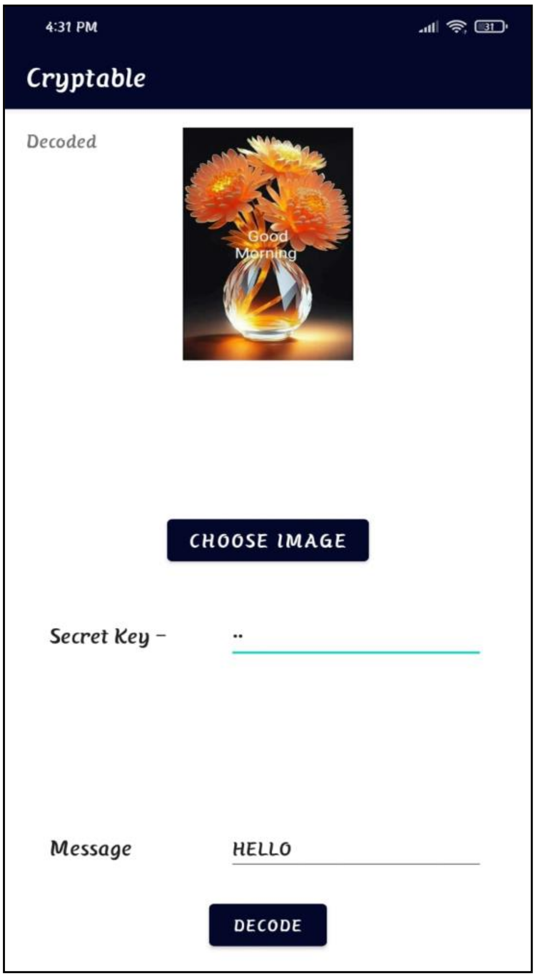
Steganography





Message – HELLO

Secret Key – ··



Secret Key – ··

Message HELLO

4. Technology/Language/Development Tools/Hardware

- **Hardware Requirement:**

  Desktop/laptop
  Minimum Ram: - 4GB
  Recommended Ram: - 8Gb or higher

- **Software requirement:**

  operating system(windows 7,8,10, mac, linux)
  software: android studio.

5. Conclusion & Future Scope

- The app will make everyone’s life easier. This is a very useful app in our society in the communication field.
- This will bring a drastic change in the field of communication.
- In this project, we deal with the concepts of security of digital data communication across the network and is designed using cryptography and steganography for data hiding.
- It is designed by using the steganography and cryptography features for better performance.
- As technology is increasing day by day there is much need for such techniques for security in future.

6. References/Resource Material/Data collection

- file:///D:/College%20Documents(sem%201,2,3,4,5)/sem%205%20Documents/pdf/ins.pdf
- https://www.techtarget.com/searchsecurity/definition/cryptography
- -https://purplesec.us/resources/cyber-security-statistics/
- -https://www.hindawi.com/journals/scn/2017/5397082/
- -https://www.geeksforgeeks.org/need-of-information-security/
- -https://www.geeksforgeeks.org/difference-between-aes-and-des-ciphers/
- https://commons.erau.edu/cgi/viewcontent.cgi?article=1039&context=jdfsl#:~:text=A%20special%20problem%20for%20both,of%20attack%20and%20signal%20modification
- -https://code-projects.org/digital-stegano-raphy-project-report-in-java-netbeans-ide-and-mysql-free-download/