SOMAIYA VIDYAVIHAR UNIVERSITY

K. J. Somaiya College of Engineering, Mumbai -77

(A Constituent College of Somaiya Vidyavihar University

Somaiya Vidhyavihar University

Encryption Software

Submitted at the end of semester IV in partial fulfilment of requirements

Of Bachelors in Technology in Computer Engineering

by

Palak Chopra
16010120008

Sanyukta Joshi
16010120019

Guide
Mr. Ninad Mehendale

SOMAIYA VIDYAVIHAR UNIVERSITY
K. J. Somaiya College of Engineering, Mumbai -77
(A Constituent College of Somaiya Vidyavihar University

Department of Computer Engineering
K. J. Somaiya College of Engineering, Mumbai-77
(Autonomous College Affiliated to University of Mumbai)
Batch 2020 - 2024

K. J. Somaiya College of Engineering, Mumbai-77

Certificate

This is to certify that the MINIPROJECT report entitled Phishing Detection submitted by Palak Chopra and Sanyukta Joshi at the end of semester IV of SY B. Tech are Bonafide record for partial fulfilment of requirements for the degree of Bachelors in Computer Engineering of University of Mumbai

Guide **Head of the Department**

Date: 7-05-2022
Place: Mumbai-77

SOMAIYA VIDYAVIHAR UNIVERSITY
K. J. Somaiya College of Engineering, Mumbai -77
(A Constituent College of Somaiya Vidyavihar University

K. J. Somaiya College of Engineering, Mumbai-77
(Autonomous College Affiliated to University of Mumbai)

Certificate of Approval of Examiners

We certify that this Mini Project report entitled Phishing Detection is bona fide record of Mini project work done by Palash H. Panchal, Rahi N. Patil, Apurva S. Rasal during semester IV.
This Mini project work is submitted at the end of semester IV in partial fulfillment of requirements for the degree of Bachelors in Technology in Computer Engineering of University of Mumbai.

Internal Examiner 1	**Internal Examiner 2**

Date:

Place: Mumbai-77

SOMAIYA VIDYAVIHAR UNIVERSITY
K. J. Somaiya College of Engineering, Mumbai -77
(A Constituent College of Somaiya Vidyavihar University

DECLARATION

We declare that this written report submission represents the work done based on our and / or others' ideas with adequately cited and referenced the original source. We also declare that we have adhered to all principles of intellectual property, academic honesty and integrity as we have not misinterpreted or fabricated or falsified any idea/data/fact/source/original work/ matter in my submission.

We understand that any violation of the above will be cause for disciplinary action by the college and may evoke the penal action from the sources which have not been properly cited or from whom proper permission is not sought.

Signature of the Student	**Signature of the Student**
16010120008	**16010120008**

Date:

Place: Mumbai-77

Introduction

There is a need for protection and confidentiality of digital data either stored on computer
systems or transmitted through a network. Encryption and decryption helps protect private information, sensitive data, and can enhance the security of communication between client apps and servers. When data is encrypted, even if an unauthorized person
if an entity gains access to it, they will not be able to read it.
In this project, we created a software to encrypt and decrypt files using two different algorithms.
The first algorithm is a custom algorithm we formulated for image encryption at the pixel level.
The second algorithm uses pre-existing cryptographic algorithms for image encryption at a byte level.
We also created a user-friendly, easy to interact with User Interface.
This report discusses the theoretical concepts applied, the algorithm used and the source code of the project.

Literature survey

Keyword - cybersecurity, encryption, decryption, cryptography

1. Attacks against files
   Employee data theft: A case study [1]
   Gillware Digital Forensics was approached with a data theft case where a company was in civil lawsuit with a former employee who took the company's proprietary data and used it to go into business themself.
   The employee, before leaving, took their company laptop home with them.
   On examination of the laptop, it was found that the files in the laptop were modified, accessed and created right until the day it was sent to the forensic experts. They traced back the footprints to resurrect the previous history of the relevant files.
   All this evidence proved that the employee had stolen the company's data.

2. Necessity of file encryption [2]
   The above case study shows why we need file and data encryption.
   Encrypting the file will protect its contents from being read by anyone who doesn't have the encryption key.
   An encryption key in the form of a password or passphrase is used to transform the file's contents in a way to make it unreadable.

3. Existing file security systems [3]
   a. Password protecting
      Word, excel and pdfs can be password protected. Other file types can be zipped and can be applied a password to.

   b. AES
      This algorithm is a specification for the encryption of electronic data established by the U.S National Institute of Standards and Technology (NIST) in 2001.

AES is a block cipher whose key size can be 128/192/256 bits and encrypts data in blocks of 128 bits each in 10/12/14 rounds, depending on the key size.

Project design

System design:

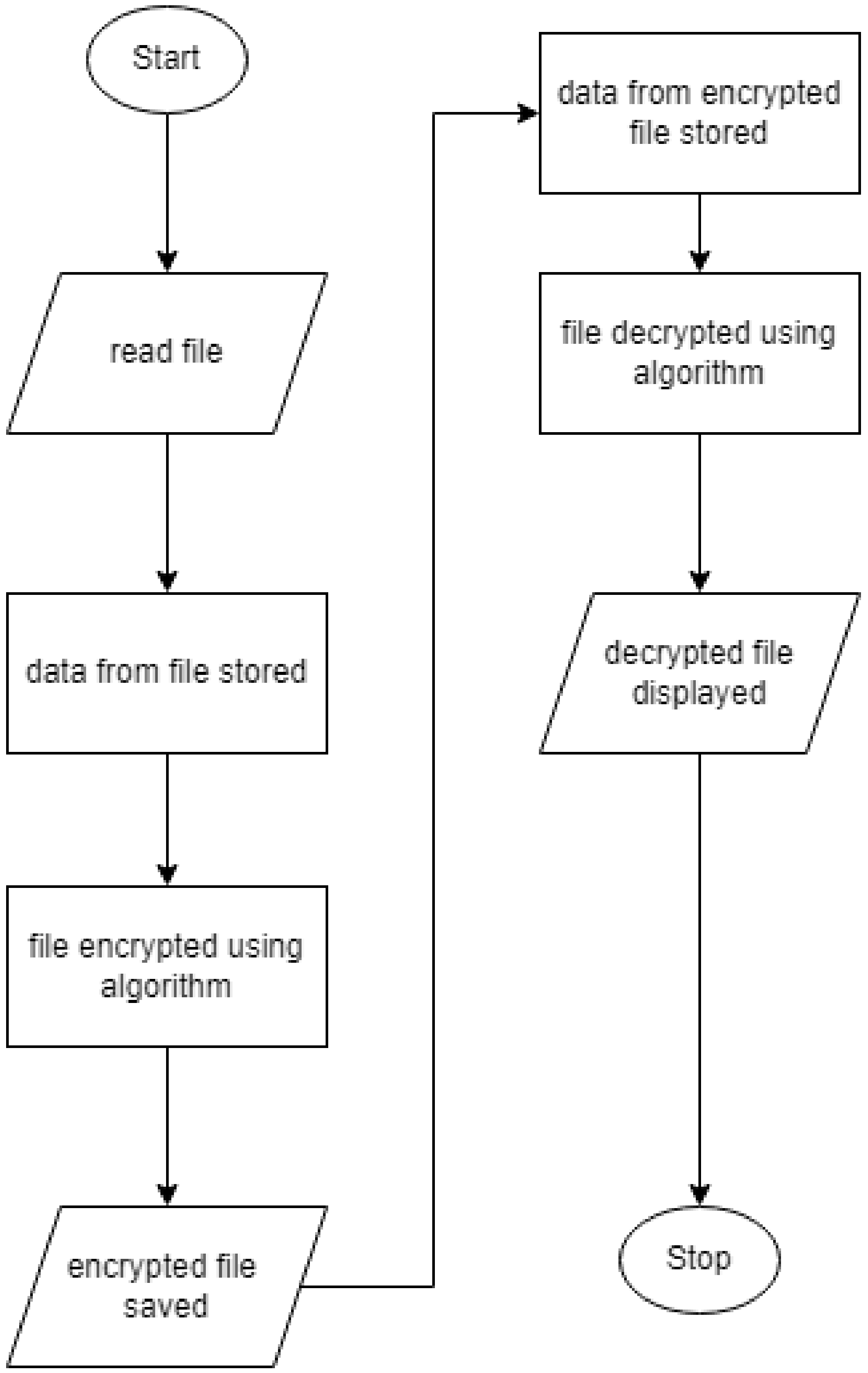
The system is divided as so:

1. The custom algorithm: this uses the formulated algorithm for encryption and decryption of an image.
2. Traditional algorithm: this uses traditional encryption methods to encrypt all kinds of files.
3. GUI: this provides and interactive user interface for the system.

System architecture:

A GUI is used to give user a selection between encrypting and decrypting a file. They then upload the file.
On choosing our custom algorithm, they get to generate a random key. The result is an encrypted image. Same method is followed for the decryption of the image using this algorithm.
On choosing the classic algorithm, the file is given an encrypted format. The user has to simply upload this image to get the decrypted result.

Software design document


Start
read file
data from file stored
file encrypted using algorithm
encrypted file saved
data from encrypted file stored
file decrypted using algorithm
decrypted file displayed
Stop

Project management plan:
The work was divided amongst the group members on the basis of front-end and back-end.

Member 1:
Palak Chopra
Worked on the back-end. This involved formulating the custom algorithm and implementing it in Java.

Member 2:
Sanyukta Joshi
Worked on the front-end. This involved making the GUI using JavaFX and connecting the main code to it.

Tools:
Programming languages:
Java:
Java is object-oriented. This allows us to create modular programs and reusable code.
Java is platform-independent: One of the most significant advantages of Java is its ability to move easily from one computer system to another. The ability to run the same program on many different systems is crucial to World Wide Web software, and Java succeeds at this by being platform-independent at both the source and binary levels.
Because of Java's robustness, ease of use, cross-platform capabilities and security features, it is the language of our choice.

Other development tools:
This project is a desktop application and so we have used JAVAFX for the UI. A myriad
of JavaFX libraries were used to achieve a user- friendly interface.

Libraries

BufferedImage - The BufferedImage subclass gives us an Image with an accessible buffer of image data.

ImageIO – This library allows us to perform image encoding and decoding.
File – This class allows us to work with files.

Random – It is used to generate a stream of pseudorandom numbers.

Color – It is used to encapsulate colors in the default sRGB color space or colors in arbitrary color spaces identified by a ColorSpace.

Implementation:

Functions:

keygen:
The keygen function used random to create a random integer string of length 1024.
encrypt:
The encrypt traversed through the image pixel-by-pixel and obtained the rgb value. It
also used a temporary key which was the subsequent 3 digits of the main key.
If the sum of the row and column of the pixel was divisible by 2, the rgb array was shifted
to assign values brg.
If the sum of the row and column of the pixel was divisible by 4, the rgb values were
stored in an array and then reversed.
If the sum of the row and column of the pixel was not divisible by 2, the temporary key
was added to the rgb values and then a mod of 256 was taken.

decrypt:
This function is the reverse of the encrypt function.
If the sum of the row and column of the pixel was divisible by 2, the rgb array was shifted
to assign values gbr.
If the sum of the row and column of the pixel was divisible by 4, the rgb values were
stored in an array and then reversed.

If the sum of the row and column of the pixel was not divisible by 2, the temporary key
was subtracted from the rgb values and then a mod of 256 was taken.

WriteImage
This function uses the ImageIO and File library to write the manipulated image to a file.

User interface:
This project is a desktop application and so we have used JAVAFX for the UI. A myriad
of JAVAFX libraries were used to achieve a user friendly interface.

CODE:

Encrypt file:

```
package endec;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.Random;

public class encryptFile
{
	private static encryptFile encrypter = new encryptFile();

	private static boolean deleteOriginal;

	private encryptFile()
	{
	}

	public static encryptFile getEncrypter(boolean originalFileDeleted)
	{
		deleteOriginal = originalFileDeleted;

		return encrypter;
	}
	public void encrypt(String srcpath, String dstpath)
	{

		System.out.print(srcpath+"\n");
		File src = new File(srcpath);

		System.out.print(dstpath+"\n");
		File dst = new File(dstpath);
		if (!dst.exists())
			dst.mkdir();
		if (!dst.isDirectory())
			return;

		try
		{
			if (!src.isDirectory())
			{
				copyEncrypted(src, dst);
			} else
			{
				File[] files = src.listFiles();

				System.out.println("Encrypting...");

				for (File f : files)
				{
					copyEncrypted(f, dst);
					if(deleteOriginal) f.delete();
				}

				System.out.println(files.length + " files are encrypted");
			}
		} catch (IOException e)
		{
			e.printStackTrace();
		}
	}

	public void copyEncrypted(File source, File dest) throws IOException
	{
		InputStream is = null;
		OutputStream os = null;

		dest = new File(dest.getPath().concat("/").concat(getRandomName(10,
"enc")));
```

```
        try
        {
            is = new FileInputStream(source);
            os = new FileOutputStream(dest);

            os.write(new byte[] { (byte) source.getName().length() });
            os.write(stringToByte(source.getName()));

            byte[] buffer = new byte[1024];

            int length;

            while ((length = is.read(buffer)) > 0)
            {
                encryptBytes(buffer);
                os.write(buffer, 0, length);
            }

        } finally
        {
            is.close();
            os.close();
        }
    }

    private void encryptBytes(byte[] data) // Encryption Algorithm is written into
here
    {
        for (int i = 0; i < data.length; i++)
        {
            data[i] = (byte) ~data[i];
        }
    }

    public byte[] stringToByte(String data)
    {
        char[] ca = data.toCharArray();
        byte[] res = new byte[ca.length * 2]; // Character.BYTES = 2;

        for (int i = 0; i < res.length; i++)
        {
            res[i] = (byte) ((ca[i / 2] >> (8 - (i % 2) * 8)) & 0xff);
        }

        return res;
    }
```

```
	public String getRandomName(int length, String extend)
	{
		Random r = new Random();
		StringBuilder res = new StringBuilder();

		for (int i = 0; i < length; i++)
		{

			char c = 'a';
			int width = 'z' - 'a';

			if (r.nextInt(3) == 0)
			{
				c = 'A';
				width = 'Z' - 'A';
			}
			if (r.nextInt(3) == 1)
			{
				c = '0';
				width = '9' - '0';
			}

			res.append((char) (c + r.nextInt(width)));
		}

		res.append(".").append(extend);

		return res.toString();
	}

	public void copy(File source, File dest) throws IOException
	{
		InputStream is = null;
		OutputStream os = null;

		try
		{
			dest = new
File(dest.getPath().concat("/").concat(source.getName()));

			is = new FileInputStream(source);
			os = new FileOutputStream(dest);

			byte[] buffer = new byte[1024];
```

```
				int length;
				int tl = 0;

				while ((length = is.read(buffer)) > 0)
				{
					tl += length;
					os.write(buffer, 0, length);
				}

				System.out.println(tl + " bytes");
			} finally
			{
				is.close();
				os.close();
			}
		}
}
```

Decrypt File:

```
package endec;

import java.awt.image.BufferedImage;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;

import javax.imageio.ImageIO;


public class decryptFile
{
	private static decryptFile decrypter = new decryptFile();

	private static boolean deleteOriginal;

	private decryptFile()
	{
	}

	public static decryptFile getDecrypter(boolean originalFileDeleted)
	{
		deleteOriginal = originalFileDeleted;
		return decrypter;
	}

	public void decrypt(String srcpath, String dstpath)
	{

		System.out.print(srcpath+"\n");
		File src = new File(srcpath);

		System.out.print(dstpath+"\n");
		File dst = new File(dstpath);

		if (!dst.exists())
			dst.mkdir();
		if (!dst.isDirectory())
			return;

		try
		{
			if (!src.isDirectory())
			{
				copyDecrypted(src, dst);
			} else
			{
				File[] files = src.listFiles();

				System.out.println("Decrypting...");

				for (File f : files)
				{
					copyDecrypted(f, dst);
					if(deleteOriginal) f.delete();
				}

				System.out.println(files.length + " files are decrytped");
			}
		} catch (IOException e)
		{
			e.printStackTrace();
		}
	}

	public void copyDecrypted(File source, File dest) throws IOException
	{
		InputStream is = null;
		OutputStream os = null;

		try
		{
			is = new FileInputStream(source);

			byte[] buffer = new byte[1024];

			byte[] name = new byte[is.read() * 2];
			is.read(name);
			String fileName = bytesToString(name);

			os = new
FileOutputStream(dest.getPath().concat("/").concat(fileName));

			int length;

			while ((length = is.read(buffer)) > 0)
			{
				decryptBytes(buffer);
				os.write(buffer, 0, length);
			}

		} finally
		{
			is.close();
			os.close();
		}
	}

	public String bytesToString(byte[] data)
	{
		StringBuilder res = new StringBuilder();

		for (int i = 0; i < data.length / 2; i++)
		{
			char c = (char) ((data[i * 2] << 8) | data[i * 2 + 1]);
			res.append(c);
		}

		return res.toString();
	}

	private void decryptBytes(byte[] data) // Decryption Algorithm is written into
here
	{
```

```
                for (int i = 0; i < data.length; i++)
                {
                        data[i] = (byte) ~data[i];
                }
        }

        public void copy(File source, File dest) throws IOException
        {
                InputStream is = null;
                OutputStream os = null;

                try
                {
                        dest = new
File(dest.getPath().concat("/").concat(source.getName()));

                        is = new FileInputStream(source);
                        os = new FileOutputStream(dest);

                        byte[] buffer = new byte[1024];

                        int length;
                        int tl = 0;

                        while ((length = is.read(buffer)) > 0)
                        {
                                tl += length;
                                os.write(buffer, 0, length);
                        }

                        System.out.println(tl + " bytes");
                } finally
                {
                        is.close();
                        os.close();
                }
        }
}
```

Encrypt Image:

```
package endec;

import java.awt.image.BufferedImage;
import java.io.IOException;
import javax.imageio.ImageIO;

import javafx.scene.image.Image;

import java.io.File;
//import java.util.Random;
import java.awt.Color;


public class encryptImage{
        public static void Encrypt(String filepath, String key) throws IOException{


                System.out.print(filepath+"\n");

                File tempFile = new File(filepath);
                BufferedImage img = ImageIO.read(tempFile);

       int width = img.getWidth();
       int height = img.getHeight();
       int nrgb, rgb, b, r, g, key_temp, k = 0, i = 0, j = 0;

       //changes
       BufferedImage img2 = new BufferedImage(
              img.getWidth(), img.getHeight(),
              BufferedImage.TYPE_INT_RGB);

       System.out.print(width + " " + height);

       int arr[][] = new int[((width*height)/2) + 1][];

       for(int row = 0; row < height; row++){
           for(int column = 0; column < width; column++){

              if(i < key.length()){
                  key_temp = Integer.parseInt(key.substring(i, i+3));
                  i+=3;
              }

              else{
                  i = 0;
```

```
    key_temp = Integer.parseInt(key.substring(i, i+3));
    i+=3;
}



rgb = img.getRGB(column, row);

Color color = new Color(rgb, true);

int red = color.getRed();
int green = color.getGreen();
int blue = color.getBlue();
if(row==6 && column==5) {
     System.out.println("BEFORE ENC: R:"+red+" G:"+green+" B:"+blue);

}

if((row+column)%2 == 0){
    nrgb = 65536 * blue + 256 * red + green;
    img.setRGB(column, row, nrgb);

    if((row+column)%4 == 0){
        arr[j] = new int[]{column, row, nrgb};
        j++;
    }

}


else{

    r = (red + key_temp) % 256;
    g = (green + key_temp) % 256;
    b = (blue + key_temp) % 256;

    if(row==6 && column==5) {
     System.out.println("key temp: "+key_temp);
     System.out.println("after ENC: R:"+r+" G:"+g+" B:"+b);

    }


    nrgb  = 65536 * r + 256 * g + b ;
    img2.setRGB(column, row, nrgb);

}
```

```
            }
        }

        System.out.println("J: ");
        System.out.println(j);

        for(int l = 0; l < j; l++){
            img.setRGB(arr[l][0], arr[l][1], arr[j-l-1][2]);
        }

        WriteImage(img2, "C:\\Users\\Sanyukta\\OneDrive - somaiya.edu\\SEMESTER -
4\\MP\\encrypted\\e.png");
    }



    public static void WriteImage(BufferedImage img, String path){
        File ImageFile = new File(path);

        try{
            ImageIO.write(img,"png", ImageFile);
        }
        catch(IOException e){
            e.printStackTrace();
        }
    }
}
```

Decrypt Image:

```
package endec;

import java.awt.image.BufferedImage;
import java.io.IOException;
import javax.imageio.ImageIO;

import javafx.scene.image.Image;

import java.io.File;
//import java.util.Random;
import java.awt.Color;


public class decryptImage{

          public static void Decrypt(String filepath, String key) throws IOException{
            System.out.print(filepath+"\n");

            File tempFile = new File(filepath);
            BufferedImage img = ImageIO.read(tempFile);

            System.out.print("\n"+key+"\n");
int width = img.getWidth();
int height = img.getHeight();
int nrgb,rgb,b,r,g, key_temp, k = 0, i = 0, j = 0;

BufferedImage img2 = new BufferedImage(
        img.getWidth(), img.getHeight(),
        BufferedImage.TYPE_INT_RGB);

int arr[][] = new int[((width*height)/2) + 1][];

System.out.print(width + " " + height);


for(int row = 0; row < height; row++){
    for(int column = 0; column < width; column++){

        if(i < key.length()){
            key_temp = Integer.parseInt(key.substring(i, i+3));
            i+=3;
        }

        else{
            i = 0;
            key_temp = Integer.parseInt(key.substring(i, i+3));
            i+=3;
        }
        if(row==5 && column==5)
            System.out.println("KEY TEMP:"+key_temp);

        rgb = img.getRGB(column, row);

        Color color = new Color(rgb, true);

        int red = color.getRed();
        int green = color.getGreen();
        int blue = color.getBlue();

        if(row==6 && column==5) {
            System.out.println("BEFORE dec: R:"+red+" G:"+green+" B:"+blue);
```

```
                }


                if((row+column)%2 == 0){
                    nrgb = 65536 * green + 256 * blue + red;
                    img.setRGB(column, row, nrgb);

                    if((row+column)%4 == 0){
                        arr[j] = new int[]{column, row, nrgb};
                        j++;
                    }
                }

                else{

                    r = ((red - key_temp)%256 + 256)%256;
                    g = ((green - key_temp)%256 + 256)%256;
                    b = ((blue - key_temp)%256 + 256)%256;


                    if(row==6 && column==5) {
                     System.out.println("key temp: "+key_temp);
                     System.out.println("after dec: R:"+r+" G:"+g+" B:"+b);

                    }

                    nrgb  = 65536 * r + 256 * g + b ;
                    img2.setRGB(column, row, nrgb);


                }
            }
        }


        for(int l = 0; l < j; l++){
            img.setRGB(arr[l][0], arr[l][1], arr[j-l-1][2]);
        }
        System.out.print("\n"+key+"\n");

        WriteImage(img2, "C://Users//Sanyukta//OneDrive - somaiya.edu//SEMESTER -
    4//MP//decrypted//d.png");

    }
    public static void WriteImage(BufferedImage img, String path){
        File ImageFile = new File(path);

        try{
            ImageIO.write(img,"png", ImageFile);
        }
        catch(IOException e){
            e.printStackTrace();
        }
    }
}
```

JAVAFX Controller Classes:

Main window:

```
package application;

import java.net.URL;
import java.util.ResourceBundle;

import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.fxml.Initializable;
import javafx.scene.Parent;
import javafx.scene.Scene;
//import javafx.scene.control.Button;
//import javafx.scene.control.ChoiceBox;
import javafx.scene.control.ComboBox;
import javafx.scene.control.Label;
import javafx.stage.Stage;


public class encryptionController implements Initializable{
        @FXML
        public ComboBox<String> combobox;
        ObservableList<String> list = FXCollections.observableArrayList("Encrypt File",
"Decrypt File");
        @FXML
        public Label mylabel;
        @FXML
        public Label l;
```

```
		@Override
		public void initialize(URL location, ResourceBundle resources) {
			// TODO Auto-generated method stub
			combobox.setItems(list);

		}
		public void comboChanged(ActionEvent event) throws Exception{
			mylabel.setText("Opening "+combobox.getValue()+" Window...");
			String message=combobox.getValue();
			if(message=="Encrypt File") {
				Stage primaryStage= new Stage();
				Parent root=
FXMLLoader.load(getClass().getResource("/application/enc.fxml"));
				Scene scene = new Scene(root,500,500);

	scene.getStylesheets().add(getClass().getResource("application.css").toExternalFor
m());
				primaryStage.setScene(scene);
				primaryStage.show();
			}
			else if(message=="Decrypt File") {
				Stage primaryStage= new Stage();
				Parent root=
FXMLLoader.load(getClass().getResource("/application/dec.fxml"));
				Scene scene = new Scene(root,500,500);

	scene.getStylesheets().add(getClass().getResource("application.css").toExternalFor
m());
				primaryStage.setScene(scene);
				primaryStage.show();
			}

		}


}
```

Encryption Window:

```
package application;

import endec.passwordgen;

import endec.encryptImage;

import java.io.File;
import java.net.URL;
import java.util.Optional;
import java.util.ResourceBundle;

import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.fxml.Initializable;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Alert;
import javafx.scene.control.Alert.AlertType;
import javafx.scene.control.Button;
import javafx.scene.control.ButtonType;
//import javafx.scene.control.Button;
//import javafx.scene.control.ChoiceBox;
import javafx.scene.control.ComboBox;
import javafx.scene.control.Label;
import javafx.scene.control.PasswordField;
import javafx.scene.image.Image;
import javafx.scene.input.Clipboard;
import javafx.scene.input.ClipboardContent;
import javafx.stage.FileChooser;
import javafx.stage.Stage;
//import javafx.scene.layout.TilePane;
//import javafx.stage.Stage;

public class encController implements Initializable{
	@FXML
	public ComboBox<String> algolist;
	ObservableList<String> LIST = FXCollections.observableArrayList("GENERAL ENC 
ALGORITHM", "CUSTOM IMAGE MANIPLUATION ALGORITHM");

	/*@FXML
	public ComboBox<String> algolist2;
```

```
ObservableList<String> LIST2 = FXCollections.observableArrayList("AES");*/

@FXML
public Label selectedAlgo;
@FXML
public Label file;

@FXML
private PasswordField passwordPasswordField;

@FXML
public Label pwd;

@FXML
Button b=new Button("Generate Password");

FileChooser fc = new FileChooser();
File selectedFile;
int flag=0;

public String message= endec.passwordgen.generateValidPassword();

public void saveFile(ActionEvent event1) {

        //File selectedFile= fc.showOpenDialog(null);
        selectedFile= fc.showOpenDialog(null);
        if(selectedFile != null) {
                file.setText("You chose: "+selectedFile.getAbsolutePath());
        }
        else {
                file.setText(null);
        }
        String fileName =selectedFile.getPath();
        String ext = fileName.substring(fileName.length()-3);
        System.out.println("\nExtension:"+ext+"\n");
        String extension = ext.toString();

if(!"png".equalsIgnoreCase(extension)&&!"jpg".equalsIgnoreCase(extension)) {
                algolist.setDisable(true);
                passwordPasswordField.setDisable(true);
        }
        else {
                algolist.setDisable(false);
                passwordPasswordField.setDisable(false);
        }
}
```

```
        public void comboAlgo(ActionEvent event2) throws Exception{
                selectedAlgo.setText("You chose: "+algolist.getValue());
            //System.out.println(algolist.getValue());
            String message=algolist.getValue();
            if(message=="GENERAL ENC ALGORITHM") {
                flag=1;
            }
            else if(message=="CUSTOM IMAGE MANIPLUATION ALGORITHM") {
                flag=2;
            }

        }
        @Override
        public void initialize(URL location, ResourceBundle resources) {
            // TODO Auto-generated method stub
            algolist.setItems(LIST);
            //algolist2.setItems(LIST2);

        }

        public void encryptBtn(ActionEvent event3) throws Exception {

            String fileName =selectedFile.getPath();

            String ext=fileName.substring(fileName.length()-3);

            System.out.print("ext:"+ext);
            System.out.println("\nExtension:"+ext+"\n");
            String ext1="png";
            String ext2="jpg";
            String extension = ext.toString();


    if(!"png".equalsIgnoreCase(extension)&&!"jpg".equalsIgnoreCase(extension)){

                System.out.println("This is running");
                endec.encryptFile en = endec.encryptFile.getEncrypter(true);
                en.encrypt(fileName, "C://Users//Sanyukta//OneDrive -
somaiya.edu//SEMESTER - 4/MP/encrypted");
            }
            else {
                if(flag==1) {
                    System.out.println("\nFlag:"+flag);
                    endec.encryptFile en =
endec.encryptFile.getEncrypter(true);
```

```
                    en.encrypt(fileName, "C://Users//Sanyukta//OneDrive
- somaiya.edu//SEMESTER - 4/MP/encrypted");
                }
                else if(flag==2) {
                    System.out.println("\nFlag:"+flag);
                    endec.encryptImage.Encrypt(fileName, message);
                }
            }


        }
        public void generatePwd(ActionEvent event) {


            final Clipboard clipboard = Clipboard.getSystemClipboard();
            final ClipboardContent content = new ClipboardContent();
            content.putString(message);
            content.putHtml(message);
            clipboard.setContent(content);
            pwd.setText("Password generated and saved to clipboard!");
    }

}
```

Decryption Window:

```
package application;

import endec.passwordgen;
import endec.decryptImage;

import java.io.File;
import java.net.URL;
import java.util.Optional;
import java.util.ResourceBundle;

import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.fxml.Initializable;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Alert;
import javafx.scene.control.Alert.AlertType;
import javafx.scene.control.ButtonType;
//import javafx.scene.control.Button;
//import javafx.scene.control.ChoiceBox;
import javafx.scene.control.ComboBox;
import javafx.scene.control.Label;
import javafx.scene.control.PasswordField;
import javafx.scene.image.Image;
import javafx.stage.FileChooser;
import javafx.stage.Stage;
//import javafx.scene.layout.TilePane;
//import javafx.stage.Stage;

public class decController {
	//@FXML
	//public ComboBox<String>
	//ObservableList<String> LIST = FXCollections.observableArrayList("AES",
"CUSTOM ALGORITHM");
	//@FXML
	//public Label selectedAlgo;
	@FXML
	public Label file;

	@FXML
	private PasswordField passwordField;


	FileChooser fc = new FileChooser();
	public File selectedFile;


	public void saveFile(ActionEvent event) {

		selectedFile= fc.showOpenDialog(null);
		if(selectedFile != null) {
			file.setText("You chose: "+selectedFile.getAbsolutePath());
		}
		else {
			file.setText(null);
		}
		String fileName =selectedFile.getPath();
		String ext=fileName.substring(fileName.length()-3);
		System.out.print("ext:"+ext);
		System.out.println("\nExtension:"+ext+"\n");
		String extension = ext.toString();
		if(!"png".equalsIgnoreCase(extension)&&!"jpg".equalsIgnoreCase(extension)) {
			System.out.println("Running");
			passwordField.setDisable(true);
		}
		}



		//@Override
		/*public void initialize(URL location, ResourceBundle resources) {
			// TODO Auto-generated method stub
			algolist.setItems(LIST);

		}*/

		public void decryptBtn(ActionEvent event) throws Exception {

			String fileName =selectedFile.getPath();
			String message = passwordField.getText();
			System.out.print(message);
			String ext=fileName.substring(fileName.length()-3);
			System.out.print("ext:"+ext);
			System.out.println("\nExtension:"+ext+"\n");
			/*if(ext=="enc") {

			}*/
			String extension = ext.toString();

	if(!"png".equalsIgnoreCase(extension)&&!"jpg".equalsIgnoreCase(extension)) {
			endec.decryptFile de = endec.decryptFile.getDecrypter(true);
			de.decrypt(fileName, "C://Users//Sanyukta//OneDrive -
somaiya.edu//SEMESTER - 4/MP/decrypted");
			System.out.println("Decrypted!");
			}
			else {
				System.out.println("Running");
				endec.decryptImage.Decrypt(fileName, message);
			}

		}
}
```

Main Class:

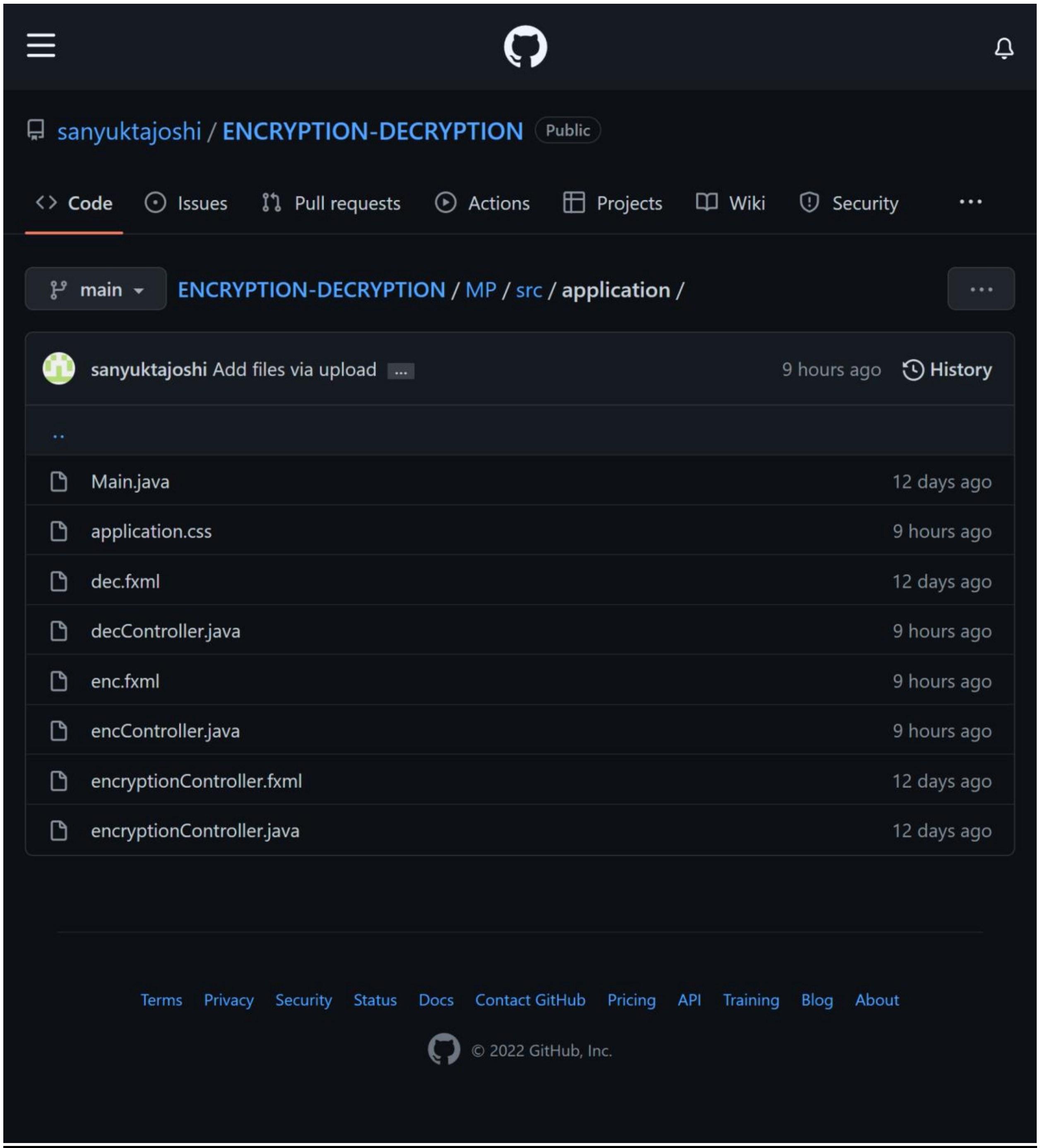
```
package application;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.stage.Stage;
import javafx.scene.Parent;
import javafx.scene.Scene;



public class Main extends Application {
	@Override
	public void start(Stage primaryStage) throws Exception {
		try {
			Parent root=
FXMLLoader.load(getClass().getResource("/application/encryptionController.fxml"));
			Scene scene = new Scene(root,500,500);

	scene.getStylesheets().add(getClass().getResource("application.css").toExternalFor
m());
			primaryStage.setScene(scene);
			primaryStage.show();
		} catch(Exception e) {
			e.printStackTrace();
		}
	}

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

Testing phases

Test case	Description	Intended result	Actual result	Completed by
Basic pixel manipulation	Basic pixel manipulation was to be done	A partially red and blue image.	An image which was half red and half blue, based its width and height.	Creating a BufferedImage object and using Color library to get and set RGB value of a pixel.
Formulation of encryption algorithm	The conditions and formulas to be used for image encryption were to be formulated	Formulas to give an encrypted RGB value and to reverse it to give the original value.	A formula which used the initial RGB value of the pixel and manipulated it based on 3 conditions, using either the key or internal or external right shift. The decryption formula was obtained by reversing the logic.	Researching Java image manipulation and standard methods of encryption. Using algebra to make formula to receive values within a range. And to also reverse the formula to obtain the original RGB value.

Applying formula to code	The formula was to be applied to actual images in code.	Actual encryption and decryption of image	The image was encrypted and decrypted as intended using the formulas. Errors were faced but debugged.	Using inbuilt Java libraries and operators to apply the theoretical formula to code.
Testing GUI	Taking input the file from the GUI.	Successful input of file and application of code to it.	The file was successfully taken input and the code was applied to it.	Using JavaFX to create the UI.

Github implementation:



sanyuktajoshi / ENCRYPTION-DECRYPTION Public

ENCRYPTION-DECRYPTION / MP / src / application /

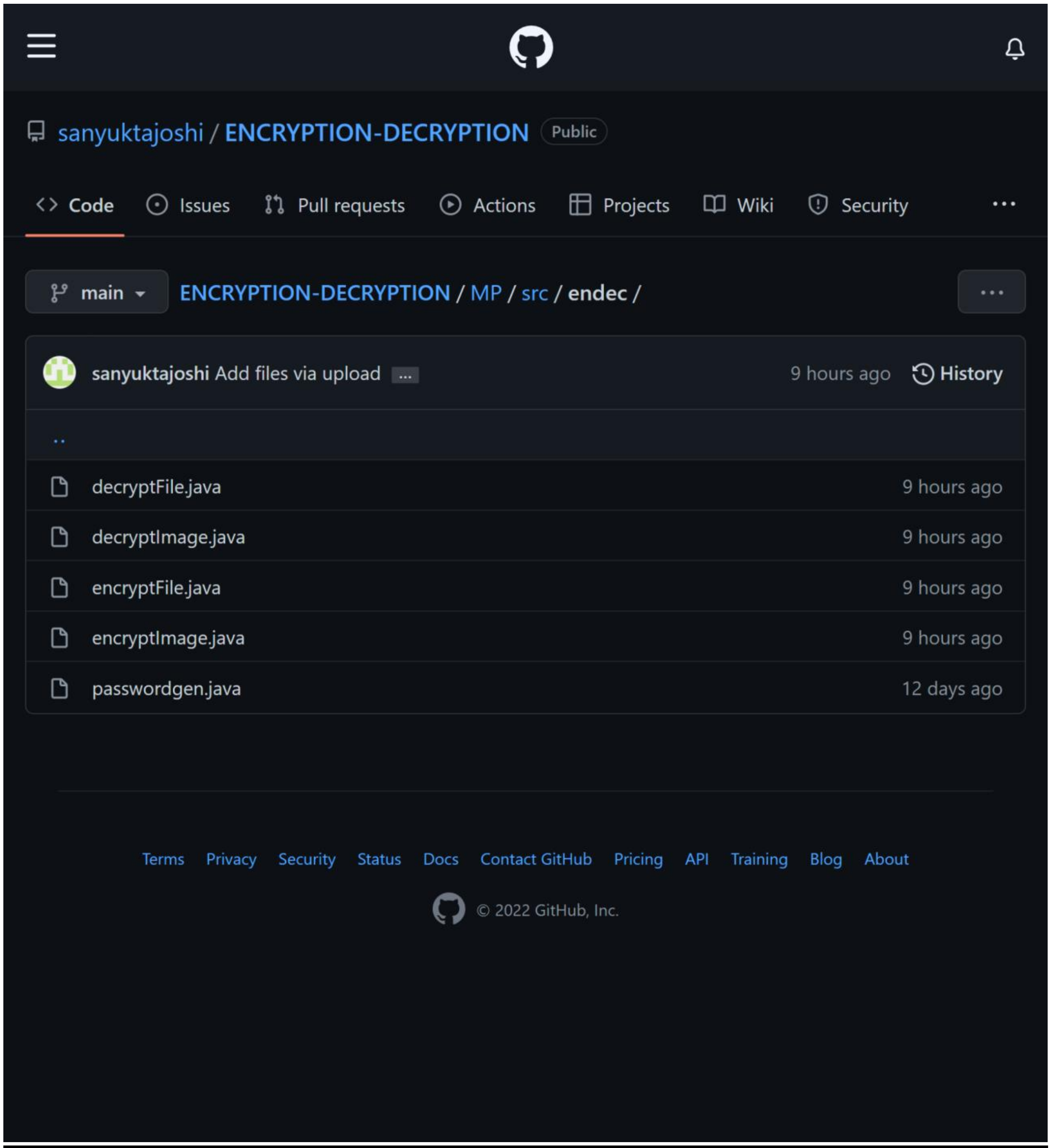
sanyuktajoshi Add files via upload ... 9 hours ago History

..

Main.java	12 days ago
application.css	9 hours ago
dec.fxml	12 days ago
decController.java	9 hours ago
enc.fxml	9 hours ago
encController.java	9 hours ago
encryptionController.fxml	12 days ago
encryptionController.java	12 days ago

Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

© 2022 GitHub, Inc.



sanyuktajoshi / ENCRYPTION-DECRYPTION Public

ENCRYPTION-DECRYPTION / MP / src / endec /

sanyuktajoshi Add files via upload ... 9 hours ago

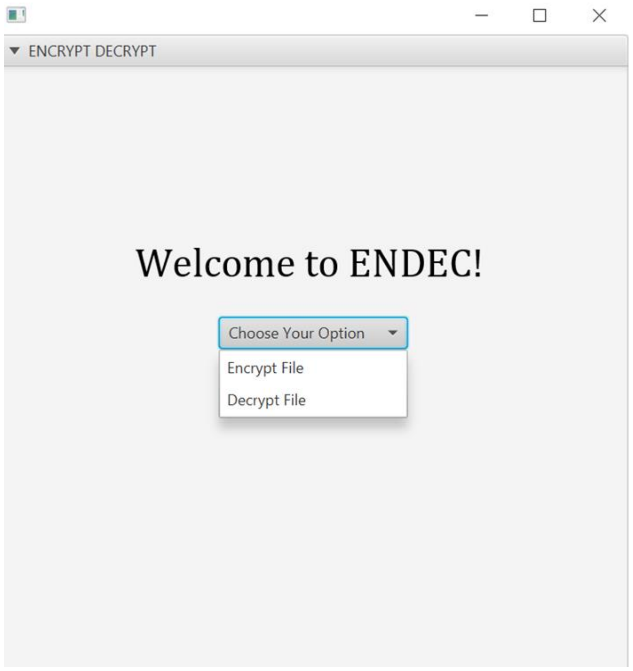
..

decryptFile.java	9 hours ago
decryptImage.java	9 hours ago
encryptFile.java	9 hours ago
encryptImage.java	9 hours ago
passwordgen.java	12 days ago

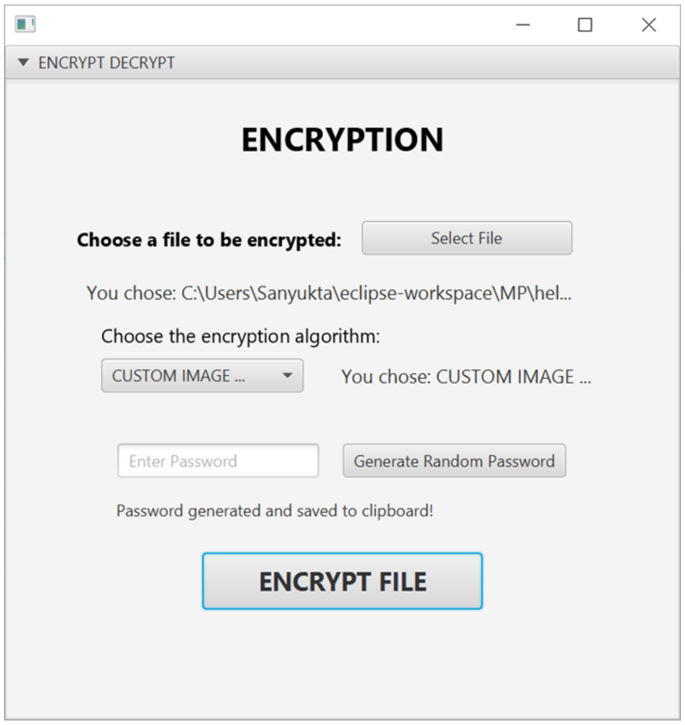
© 2022 GitHub, Inc.

Result:

Main Window:



Encryption Window:



Choose a file to be encrypted:

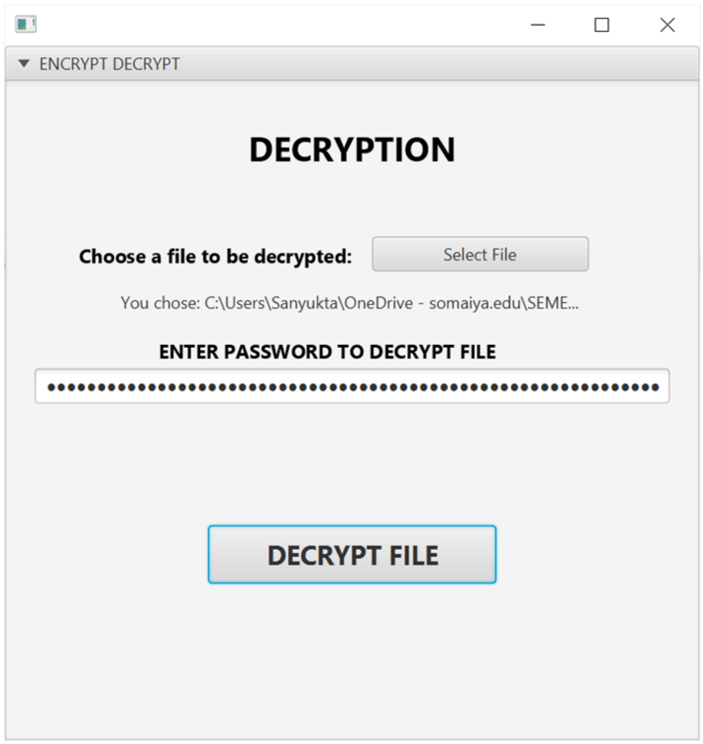
You chose: C:\Users\Sanyukta\eclipse-workspace\MP\hel...

Choose the encryption algorithm:

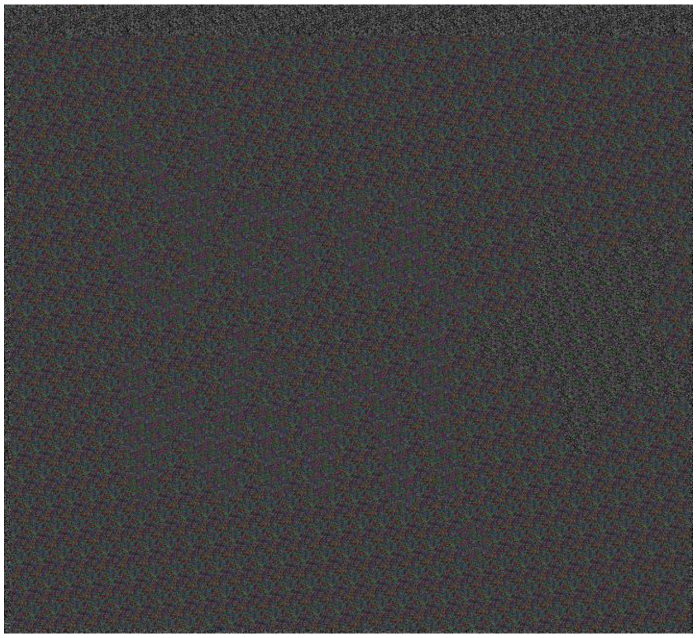
You chose: CUSTOM IMAGE ...

Password generated and saved to clipboard!

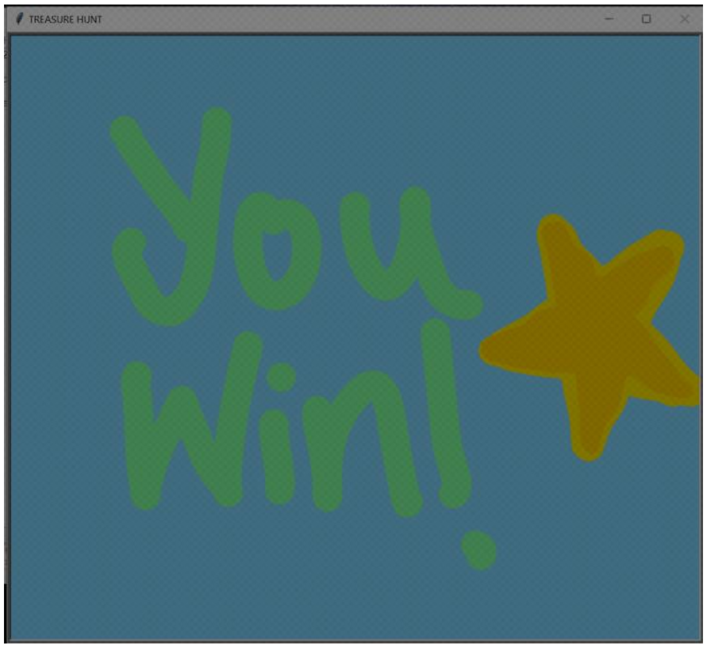
Decryption Window:


ENCRYPT DECRYPT
DECRYPTION
Choose a file to be decrypted:
Select File
You chose: C:\Users\Sanyukta\OneDrive - somaiya.edu\SEME...
ENTER PASSWORD TO DECRYPT FILE
●●
DECRYPT FILE

Encrypted image:



Decrypt image:



Conclusion

We observed the importance of file encryption through our employee-theft case study.
This can be done by using encryption softwares, one of which was created in this project.
We used existing encryption system and modulus algebra as inspiration for our algorithm that demonstrated in extreme detail how encryption and decryption works.
Our software successfully gave us our desired output and acted as a great learning opportunity of the world of cybersecurity and cryptography.

Future scope

Given the opportunity to further work on this project, features to transmit files through an encrypted channel from within the software can be made.
We can also add a feature where on a given number of incorrect decryption tries, the file gets destroyed to prevent it from brute-forcing.
Authentication methods can also be added where only certain users, chosen by the encryptor, identified from their user ID, can decrypt the file.

Reference:

1. https://www.tetradefense.com/incident-response-services/case-study-employee-data-theft/
2. https://oit.williams.edu/help-guides/device-security-virus-protection/file-encryption/
3. https://www.simplilearn.com/data-encryption-methods-article