A PROJECT REPORT

ON

"IMAGE ENCRYPTION AND DECRYPTION"

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF

DIPLOMA IN COMPUTER ENGINEERING

BY

GROUP MEMBERS:

MD ZEESHAN SIDDIQUI	**20054-CM-110**
SAI SANTHOSH	**20054-CM-094**
T. SAI VAMSHI	**20054-CM-118**
HAMILPURE ASHMITH	**20054-CM-104**
MOHAMMED ZAID AHMED	**20054-CM-087**

UNDER THE GUIDANCE OF **SRI M.AYYAPPA** SIR,
TRAINERS OF INDUSTECH SOLUTIONS PVT.LTD

GOVERNMENT INSTITUTE OF ELECTRONICS, SECUNDERABAD

(Approved by TS SBTET,Hyderabad)

East Marredpally,Secunderabad-500026

2020-2023

ACKNOWLEDGEMENT

With great pleasure and deep sense of gratitude , we acknowledge the contribution of All the faculty for the successful completion of this project.

We deem to be great honor to thank **Mr. G. Venkateshwara prasad** garu **M.Tech**, Principal **GOVERNMENT INSTITUTE OF ELECTRONICS** who is an axle behind our studies and for having given me a chance to pursue my Diploma course in this esteemed institution

. Our thanks are due to **Smt. K. Radhik**a garu **M.TECH** , Head of the department for her excellent, expert support rendered to me during my tenure in the institute. We would like to express our gratitude and appreciation to all the people who gave us the possibility to complete this report and training at IndusTech. It is always a pleasure to remind the fine people for their sincere guidance we received to uphold our practical as well as theoretical skills in this industrial training.

Finally I would like to thank all the trainers at IndusTech who gave us all the support required to understand all the technical aspects both in theory and practice in the training.

GROUP MEMBERS:

MD ZEESHAN SIDDIQUI	20054-CM-110
SAI SANTHOSH	20054-CM-094
T. SAI VAMSHI	20054-CM-118
HAMILPURE ASHMITH	20054-CM-104
MOHAMMED ZAID AHMED	20054-CM-087

DECLARATION

We hereby declare the project report entitled

"DATA ENCRYPTION AND DECRYPTION"

Is the result in training given to us

During **DECEMBER TO MAY 2022 IN**

"INDUSTECH INDUSTRIES PVT.LTD"

For

The award of **"DIPLOMA IN COMPUTER ENGINEERING"**

BY

STATE BOARD OF TECHNICAL EDUCATION AND TRAINING
HYDERABAD, TELANGANA

We hereby declare that this project was the outcome of our efforts and has not been submitted to any other university for the award of any degree or diploma.

GROUP MEMBERS:

MD ZEESHAN SIDDIQUI	20054-CM-110
SAI SANTHOSH	20054-CM-094
T. SAI VAMSHI	20054-CM-118
HAMILPURE ASHMITH	20054-CM-104
MOHAMMED ZAID AHMED	20054-CM-087

	INDEX	
Sr. No.	Description	Page No.
	Abstract	5
1	Chapter-1 Introduction 1.1 Introduction 1.2 Cryptography !.3 Encryption and Decryption 1.4 Difference between Encryption and Decryption	6-8
2	Chapter 2 Requirements 2.1 Software Required	9-11
3	Chapter-3 Review of Literature 3.1 DES Introduction 3.2 History 3.3 Overview 3.4 Architecture of DES	12-14
4	Chapter-4 Comparative Analysis 4.1 Double Data Encryption Standard 4.2 Proposed Method 4.2 Proposed Method 4.3 Triple DES and improvement over original DES	15-16
5	Chapter-5 Design Details 5.1 Why Triple Data Encryption Standards 5.2 Triple Data Encryption Standards 5.3 Architecture of Triple DES 5.4 Algorithm of TDES	17-19
6	Chapter-6 Summary 6.1 Triple DES Modes of Operation 6.2 Benefits of Using TDES 6.3 Challenges in Image Encryption and Decryption 6.4 Advantages 6.5 Disadvantages	20-21
7	Chapter-7 Code	22-
8	Chapter-8 References	
9	Chapter-9 Conclusion	

ABSTRACT

In today's world all digital services like internet communication, medical and military imaging systems, the multimedia system needs a prominent level and Protected security. There is a need for a security level to safely store and send digital images holding critical information. This is because of the faster growth in multimedia technology, the internet, cell phones. Therefore, there is a need for image encryption techniques to hide images from such attacks. In this system, we use Triple DES (Data Encryption Standard) to hide images.

Such Encryption Tech technique helps avoid Active and Passive Attacks. The tripleDES algorithm is based on The DES algorithm itself; it uses the same method as that of the DES but the difference is that it uses 3 keys rather than just one. For the encryption process, it initially encrypts the data using just one key and then decrypts the data using another different key and then finally encrypts the data again using another key. For the decryption process, it is the reverse of the encryption process: it initially decrypts the cypher data using one key, then encrypts the data using another key and then finally decrypts the data back to its original form using another different key. This algorithm uniquely defines the mathematical steps needed to transform the image into a cryptographic cipher and to transform the cipher image back to its original form.

CHAPTER 1
INTRODUCTION

1.1 Introduction to the Topic:

In this era of universal electronic connectivity, the possibility of data damage or theft is extremely high. That is why it needs time to secure data from those groups. The tremendous growth in computer systems and interconnection with networks have increased depending on the company or individual based on information stored and communicated using this system. There is a need to protect the data from disclosure and to protect systems from network-based attacks.

1.2 Cryptography:

Cryptography is a technique which is intended to transform the data and can be used to supply various security related concepts such as confidentiality, data integrity, authentication, authorization, and non-repudiation. Securing the information and other services is an important thing by using the security mechanism we must protect from unintended or unauthorized access, change or destruction. Cryptography is the art of secret writing to hide information secret or keeping a message secure. A secure network must have integrity, so that all the information stored is always correct and protected without any redundant data. Which are used to reduce network threats.

Encryption/decryption are the fundamental function of cryptography, which is used to hide the information from unauthorized users so that chances of threats are also reduced. The aim of many cryptosystems is to make their data computationally infeasible to crack by intruders. It can supply integrity as it can be used to detect any changes which may have happened to the data, and it can supply accountability as it can be used to verify the origin of the data. In

encryption a simple message (the plaintext) is converted into an unreadable form called ciphertext (scrambled message after encryption). While decryption the cipher text is converted into plain text (original form) Many encryption algorithms are widely available and used in information security.

1.3 Encryption and Decryption:

Encryption is a process which transforms the original information into an unrecognizable form. This new form of the message is entirely different from the original message. That is why a hacker is not able to read the data as senders use an encryption algorithm. Encryption is usually done using key algorithms. Data is encrypted to make it safe from stealing. However, many known companies also encrypt data to keep their trade secret from their competitors. Decryption is a process of converting encoded/encrypted data into a form that is readable and understood by a human or a computer. This method is performed by unencrypting the text manually or by using keys used to encrypt the original data.

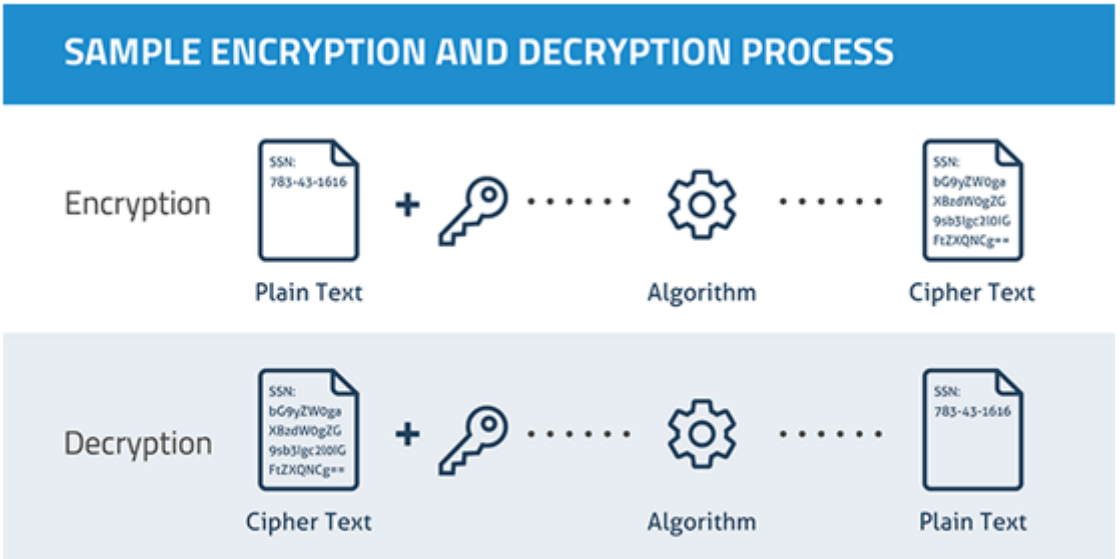


Fig 1.3 Encryption and Decryption Process

1.4 Difference between Encryption and Decryption:

Sr. no.	Decryption	Decryption
1.	Encryption is the process of converting normal messages into meaningless messages.	While decryption is the process of converting meaningless messages into its original form.
2.	Encryption is the process which takes place at the sender's end.	While decryption is the process which takes place at the receiver's end.
3.	Its major task is to convert the plain text into cipher text.	While its main task is to convert the cipher text into plain text.
4.	Any message can be encrypted with either secret key or public key.	Whereas the encrypted message can be decrypted with either secret key or private key.
5.	In the encryption process, the sender sends the data to the receiver after it is encrypted.	Whereas in the decryption process, receiver receives the information (Cipher text) and convert into plain text.

CHAPTER-2
SOFTWARE REQUIRED

Software Requirements:

Since this is a software hence it will have to run on some hardware and operating system obviously, so below are the requirements to run this software :

1. Windows/Linux/Mac OS any version, hence it can run on any platform.
2. Python3, it needs python to be installed in your system to run this successfully.
3. Packages in python -
   - Crypto.Cipher
   - Crypto.Hash
   - Tkinter
   - Turtle
   - Requests

Hardware Requirements:

In terms of hardware requirements there is not much required at all but still below requirements are must :

- Working PC or Laptop

TECHNICAL DETAILS

For this project we have used various latest technologies which will be evaluated in this chapter with every detail of why it is used.
We'll divide this section of explanation of technology based on modules/features in the project. But first let's see the language used in this project.

Language Used:

We have used **Python language** as it is very new and also comes with so many features like we can do Machine Learning, Computer Vision and Also make GUI applications with ease.

Reasons for Selecting this language :

1 – Short and Concise Language.
2 – Easy to Learn and use.
3 – Good Technical support over Internet
4 – Many packages for different tasks.
5 – Run on Any Platform.
6 – Modern and OOP language

Well these are just the minor points from our sides. Python is just a lot more than this.
Some more notes about python,

Python is a widely used general-purpose, high level programming language. It was created by Guido van Rossum in 1991 and further developed by the Python Software Foundation. It was designed with an emphasis on code readability, and its syntax allows programmers to express their concepts in fewer lines of code. Python is a programming language that lets you work quickly and integrate systems more efficiently.There are two major Python versions: **Python 2** and **Python 3**

Some specific features of Python are as follows:

- An interpreted (as opposed to compiled) language. Contrary to e.g. C or Fortran, one does not compile Python code before executing it. In addition, Python can be used interactively: many Python interpreters are available, from which commands and scripts can be executed.
- A free software released under an open-source license: Python can be used and distributed free of charge, even for building commercial software.
- Multi-platform: Python is available for all major operating systems, Windows, Linux/Unix, MacOS X, most likely your mobile phone OS, etc.
- A very readable language with clear non-verbose syntax.
- A language for which a large variety of high-quality packages are available for various applications, from web frameworks to scientific computing.
- A language very easy to interface with other languages, in particular C and C++.
- Some other features of the language are illustrated just below. For example, Python is an object-oriented language, with dynamic typing (the same variable can contain objects of different types during the course of a program)

CHAPTER 3

REVIEW OF LITERATURE

3.1 DES Introduction:

DES (Data Encryption Standard), was the first encryption standard to be recommended by NIST (National Institute of Standards and Technology). It is based on the IBM proposed algorithm called Lucifer. DES became a standard in 1974. Since that time, many attacks and methods have been recorded that exploit the weaknesses of DES, which made it an insecure block cipher. It has a key size of 56 bits. The problem with DES is that it has only 256 combinations.

Double DES which applies the algorithm twice to the plain text with a different key each time.

C=Enc (K2, Enc (K1, P))

P=Dec (K1, Dec (K2, C))

It has 2112 combinations. Due to the MITM attack (Man in the middle) it lowers the attack complexity of finding the key easily, so the attacker can find out keys in less time. It has O (256).

3.2 History:

In 1973, NIST published a request for proposals for a national symmetric-key cryptosystem. A proposal from IBM, a modification of a project called Lucifer, was accepted as DES. DES was published in the Federal Register in March 1975 as a draft of the Federal Information Processing Standard (FIPS).

After the publication, the draft was criticized severely for two reasons. First, critics questioned the small key length (only 56 bits), which could make the cypher vulnerable to a brute-force attack. Second, critics were concerned about some hidden design behind the internal structure of DES. They were suspicious that some part of the structure (the S-boxes) may have some hidden trapdoor that would allow the National Security Agency (NSA) to decrypt

the messages without the need for the key. Later IBM designers mentioned that the internal structure was designed to prevent differential cryptanalysis.

DES was finally published as FIPS 46 in the Federal Register in January 1977. NIST, however, defines DES as the standard for use in unclassified applications. DES has been the most widely used symmetric-key block cypher since its publication. NIST later issued a new standard (FIPS 46-3) that recommends the use of triple-DES (repeated DES cipher three times) for future applications. As we will see in Chapter 7, AES, the recent standard, is supposed to replace DES in the long run.

3.3 Overview:

DES is a block cipher, as shown in Fig.

Fig. 2.3 Encryption and decryption with DES

At the encryption site, DES takes a 64-bit plaintext and creates a 64-bit ciphertext; at the decryption site, DES takes a 64-bit ciphertext and creates a 64-bit block of plaintext. The same 56-bit cipher key is used for both encryption and decryption.

3.4 ARCHITECTURE OF DES :

Let us concentrate on encryption; later we will discuss decryption. The encryption process is made of two permutations (P-boxes), which we call initial and final permutations, and sixteen Feistel rounds. Each round uses a different 48-bit round key generated from the cipher key according to a predefined algorithm described later in the chapter. Figure 2.4 shows the elements of DES cipher at the encryption site.

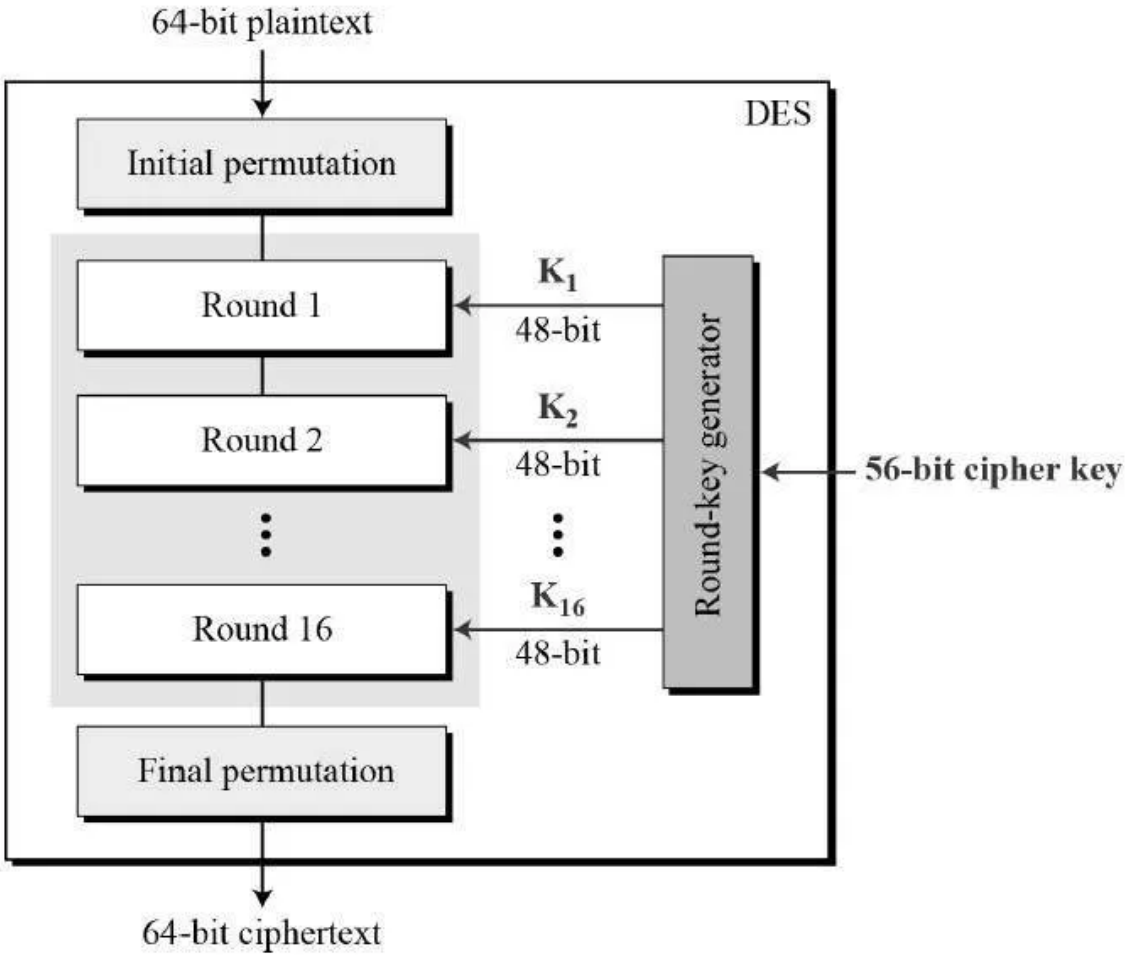


Fig. 2.4 General structure of DES

CHAPTER 4
COMPARATIVE ANALYSIS

4.1 Double Data Encryption Standard :

Double DES is an encryption technique that uses two instances of DES on the same plain text. In both instances, it uses different keys to encrypt the plain text. Both keys are needed at the time of decryption. The 64-bit plain text goes into the first DES instance which is then converted into a 64-bit middle text using the first key and then it goes to the second DES instance which gives 64-bit ciphertext by using the second key.

However double DES uses a 112 bits key but gives security level of 2^56 not 2^112 and this is because of meet-in-the middle attack which can be used to break through double DES.

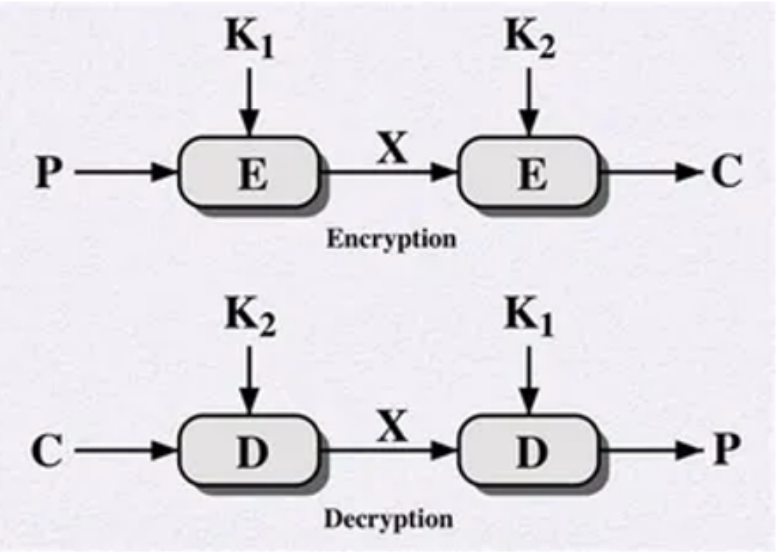


Fig 3.1 ARCHITECTURE OF Double DES

4.2 Proposed Method:

An enhancement of DES and Double DES, the 3DES (Triple DES) encryption standard was proposed. In this standard the encryption method is like the one in original DES but applied 3 times to increase the encryption level. Triple DES systems are significantly more secure than single DES, but these are clearly a much slower process than encryption using single DES.

4.3 Triple DES and improvement over original DES:

Triple DES (3DES / 3DEA) uses 3 keys of 64-bits each, with an effective key length of 56 bits (8 bits are used for parity checking). 3DES uses a block size of 64-bits.

There are 3 keying modes of 3DES:

1. Three independent keys: Wherein none of the keys are explicitly similar: K1≠K2, K2≠K3, K3≠K1. This is also known as 3TDEA.
2. Two independent keys: Wherein K1 and K2 are explicitly different but K1 and K3 are similar. K1≠K2, K2≠K3, K3=K1. This is known as 2TDEA.
3. All keys are the same. K1 = K2 = K3. This key mode when used with EDE mode of 3DES is in essence – DES.

There are two different operating modes of 3DES:
1. EDE mode: Which functions as encrypt, decrypt, encrypt with K1, K2, K3, respectively.

This mode can be denoted as: C = EK1(DK2(EK3(P))). This is the used mode as it supports backward compatibility with the original DES standard when used with Key option 2. Using key option 3 with EDE mode is in-fact original DES standard.
2. EEE mode: Wherein plaintext is encrypted 3 times using 3 keys. This is represented as: C = EK1(EK2(EK3(P)))

CHAPTER 5
DESIGN DETAILS

5.1 Why Triple Data Encryption Standards:

DES is an acronym for Data Encryption Standards. It is a technique to encrypt any plain text using a 56-bit key. Advancements in technology have led to the development of new methods that can easily crack a DES-encrypted text.

To prevent this, another encryption technique named Triple-DES was introduced. This method is much more secure than the original DES, and it uses a 168-bit key.

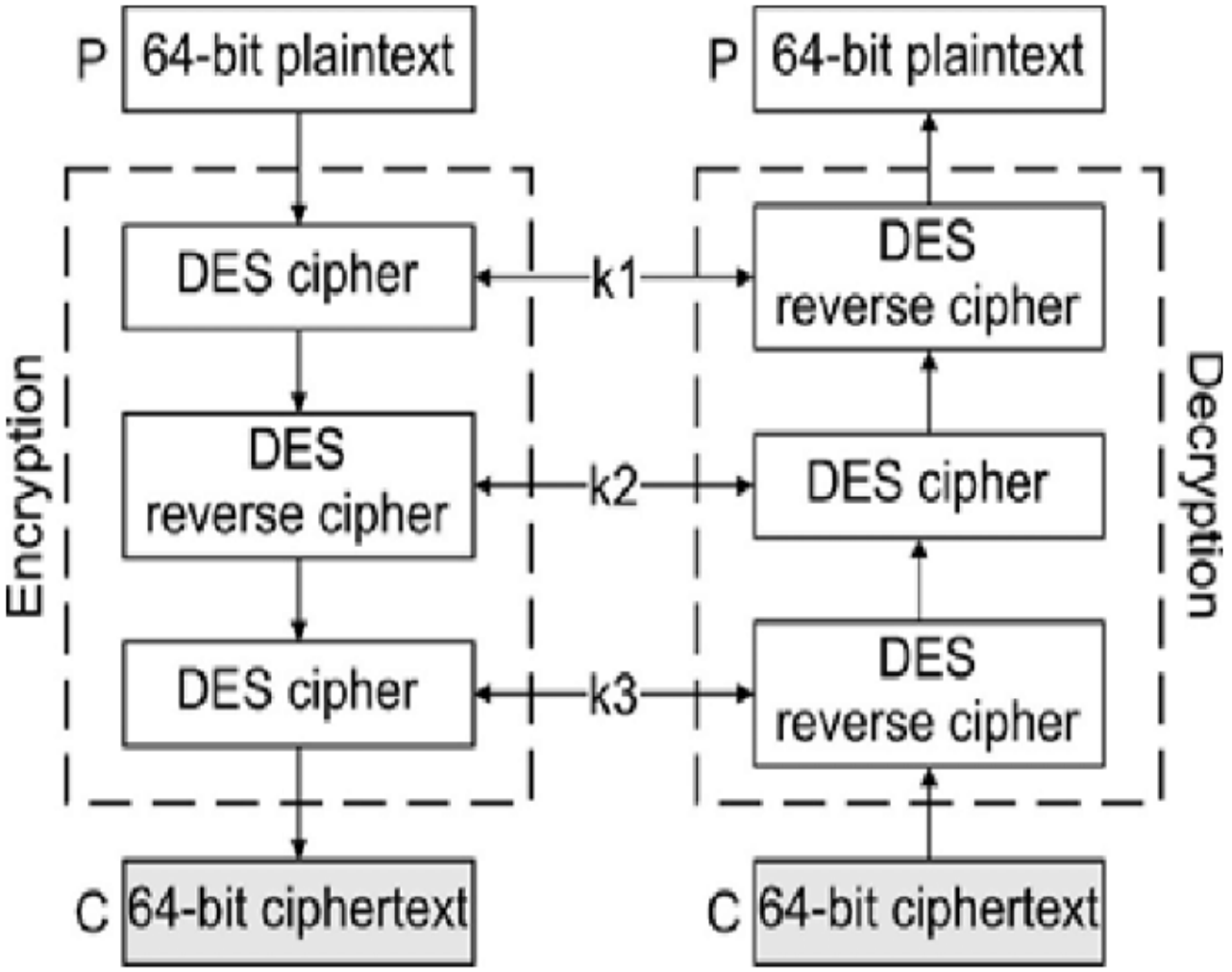
5.2 Triple Data Encryption Standards:

Triple-DES is a process in which we encrypt an image, text or video using 56 bit two keys or 128-bit keys. This kind of process may be secure but still has its flaws. To overcome this flaw, we encrypted out file with three 56-bit keys instead of two keys. Hence making it more secure. In the earlier referred case study, there was only two keys were used for encryption process. Triple-DES process follows EDE (Encryption, Decryption, Encryption) model. EDE model states that every file or text must be encrypted twice and decrypted once in a sequential order to perform encryption process. First it encrypts using one secret key and then decrypted using a different secret key finally encrypts using same encrypt key. So, the flaw was if the hacker got to know one secret key it is extremely easy to apply brute force attack. Hence to overcome this flaw we are using three different keys for every EDE process. EDE uses 192-bit keys out of which only 168 bits are used for encryption process.

Still, a strong algorithm even though we do not use the last eight bits. Which makes it more secure over a network. As the security weaknesses of DES became clearer, 3DES was proposed as a way of extending its key size without

having to build an entirely new algorithm. Rather than using a single key as in DES, 3DES runs the DES algorithm three times, with three 56- bit keys:
Key one is used to encrypt the plaintext.
Key two is used to decrypt the text that had been encrypted by key one.
Key three is used to encrypt the text that was decrypted by key two.

5.3 ARCHITECTURE OF Triple DES :



5.4 Algorithm of TDES :

Step1: Choose Encryption || Decryption
Step2: Opening file name and image
Step3: Give password as Key
Step4:

Cipher Text = EK3(DK2(EK1(plain text)))
DES encrypts with K1, DES decrypts with K2, DES encrypts with K3.
Plain Text = DK1(EK2(DK3(ciphertext)))
DES decrypt with K3, encrypt with K2, then decrypt with K1.

Chapter 6
SUMMARY

6.1 Triple DES Modes of Operation:

Experts using TDES have five different modes of operation to choose from.

1. Electronic Codebook (ECB). Each 64-bit block is encrypted and decrypted independently
2. Cipher Block Chaining (CBC). Each 64-bit block depends on the earlier one and uses an Initialization Vector (IV)
3. Cipher Feedback (CFB). The preceding ciphertext becomes the input for the encryption algorithm, producing pseudorandom output, which in turn is XORed with plaintext, building the next ciphertext unit
4. Output Feedback (OFB). Much like CFB, except that the encryption algorithm input is the output from the preceding DES
5. Counter (CTR). Each plaintext block is XORed with an encrypted counter. The counter is then incremented for each next block

6.2 Benefits of Using TDES for Image Encryption:

TDES offers several benefits for image encryption, including high-level security, efficient performance, and compatibility with various operating systems and platforms.

TDES can also be easily integrated with other encryption techniques, such as public key cryptography, to enhance the overall security of the image file. Additionally, TDES supports both hardware and software implementations, making it suitable for a wide range of applications.

6.3 Challenges in Image Encryption and Decryption:

One of the main challenges in image encryption and decryption is the need for high-speed processing. Images contain large amounts of data, and encryption and decryption can be time-consuming if not optimized.

Another challenge is the need to balance security with accessibility. While highly secure encryption techniques are desirable, they can also make it difficult for authorized parties to access the encrypted data.

6.4 Advantages of TDES:

The TDES algorithm is highly secure due to its use of three different keys and multiple rounds of encryption and decryption. This makes it ideal for protecting sensitive images and data.

Another advantage of the TDES algorithm is its compatibility with a wide range of devices and platforms. It can be implemented on both software and hardware platforms, making it easy to integrate into existing systems.

Chapter 7
Code

```
#Required Libraries importing
from tkinter import *
from tkinter import filedialog
from Crypto.Cipher import DES
from Crypto.Hash import SHA256
import turtle
import argon2
import requests
import os
from getpass import getpass
from Crypto.Protocol.KDF import PBKDF2

Key_length=100005
salt="$ez*&214097GDAKACNASC;LSOSSBAdjskasnmosuf!@#$^()_adsa"

# Create global variables to store password and name
key_enc = ""
path = ""

#Encrypting function
def encryptor():

    global path
    global key_enc
    image_path = os.path.basename(path)

    success_label = Label(root, text="")
    success_label.pack(padx=10, pady=15)

    success_label2 = Label(root, text="")
    success_label2.pack(padx=10, pady=15)


    #Opening the image file
    try:
        with open(path, 'rb') as imagefile:
            image=imagefile.read()

        #Padding
```

```
        while len(image)%8!=0:
            image+=b" "
    except:
        success_label.config(text="Error loading the file !")
        exit()

    #hashing original image in SHA256
    hash_of_original=SHA256.new(data=image)


    #Salting and hashing password
    key_enc=PBKDF2(key_enc,salt,48,Key_length)


    #Encrypting using triple 3 key DES
    success_label.config(text="Encrypting...")
    try:

        cipher1=DES.new(key_enc[0:8],DES.MODE_CBC,key_enc[24:32])
        ciphertext1=cipher1.encrypt(image)
        cipher2=DES.new(key_enc[8:16],DES.MODE_CBC,key_enc[32:40])
        ciphertext2=cipher2.decrypt(ciphertext1)
        cipher3=DES.new(key_enc[16:24],DES.MODE_CBC,key_enc[40:48])
        ciphertext3=cipher3.encrypt(ciphertext2)

        success_label.config(text="Encryption Successfull...")
    except:
        print(" Encryption failed...Possible causes:Library not installed
properly/low device memory/Incorrect padding or conversions")
        exit()

    #Adding hash at end of encrypted bytes
    ciphertext3+=hash_of_original.digest()


    #Saving the file encrypted
    try:
        dpath="encrypted_"+image_path
        with open(dpath, 'wb') as image_file:
                image_file.write(ciphertext3)
        success_label2.config(text="Encrypted Image Saved successfully as imagename
"+dpath)
```

```
    except:
        temp_path=input("Saving image failed!. Enter alternate name without format
to save the encrypted image. If it is still failing then check system memory")
        try:
            dpath=temp_path+path
            dpath="encrypted_"+path
            with open(dpath, 'wb') as image_file:
                    image_file.write(ciphertext3)
            print("Encrypted Image Saved successfully as imagename in the same
directory "+dpath)
            exit()
        except:
            print(" Failed....Exiting...")
            exit()

#decrypting function
def decryptor():
    global path
    global key_enc
    image_name = os.path.basename(path)

    success_label = Label(root, text="")
    success_label.pack(padx=10, pady=15)

    success_label2 = Label(root, text="")
    success_label2.pack(padx=10, pady=15)


    try:
        with open(path,'rb') as encrypted_file:
            encrypted_data_with_hash=encrypted_file.read()

    except:
        print(" Unable to read source cipher data. Make sure the image is in same
directory...Exiting...")
        exit()




    #extracting hash and cipher data without hash
    extracted_hash=encrypted_data_with_hash[-32:]
```

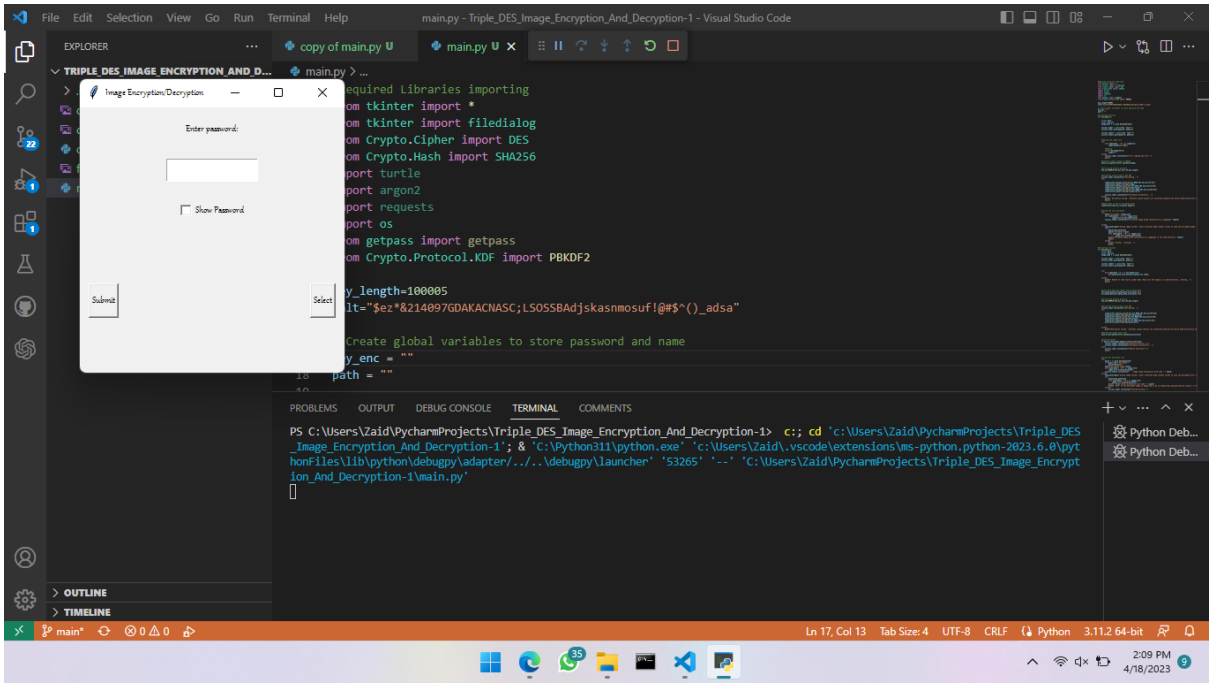
```
    encrypted_data=encrypted_data_with_hash[:-32]


    #salting and hashing password
    key_enc=PBKDF2(key_enc,salt,48,Key_length)


    #decrypting using triple 3 key DES
    success_label.config(text="Decrypting...")
    try:

        cipher1=DES.new(key_enc[16:24],DES.MODE_CBC,key_enc[40:48])
        plaintext1=cipher1.decrypt(encrypted_data)
        cipher2=DES.new(key_enc[8:16],DES.MODE_CBC,key_enc[32:40])
        plaintext2=cipher2.encrypt(plaintext1)
        cipher3=DES.new(key_enc[0:8],DES.MODE_CBC,key_enc[24:32])
        plaintext3=cipher3.decrypt(plaintext2)


    except:
        print("Decryption failed...Possible causes:Library not installed
properly/low device memory/Incorrect padding or conversions")

    #hashing decrypted plain text
    hash_of_decrypted=SHA256.new(data=plaintext3)


    #matching hashes
    if hash_of_decrypted.digest()==extracted_hash:
        success_label2.config(text="Password Correct !!!")
        success_label2.config(text="Decryption Successfull...")
    else:
        success_label.config(text="Pasword Incorrect!!!")
        exit()



    #saving the decrypted file
    try:
        epath = os.path.basename(path)
        if epath[:10]=="encrypted_":
            epath=epath[10:]
        epath="decrypted_image"+epath
```

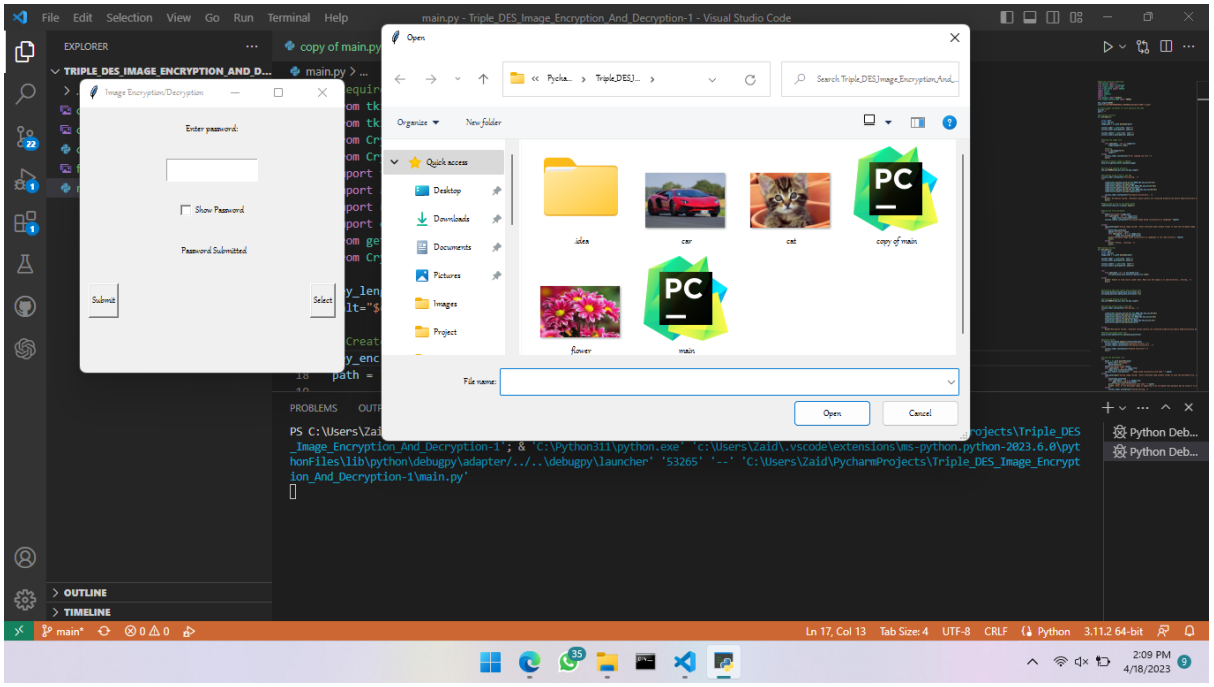
```
        with open(epath, 'wb') as image_file:
            image_file.write(plaintext3)
        success_label2.config(text="    Image saved successully with name " +
epath)
    except:
        temp_path=input("Saving image failed!. Enter alternate name without format
to save the decrypted file. If it is still failing then check system memory")
        try:
            epath=temp_path+path
            with open(epath, 'wb') as image_file:
                image_file.write(plaintext3)
            print(" Image saved successully with name " + epath)
            print(" Note: If the decrypted image is appearing to be corrupted then
password may be wrong or it may be file format error")
        except:
            success_label.config(text="Failed Exiting..")
            exit()


# Function to get password from user and store it securely
def get_key():
    global password

    password = key_entry.get()
    if show_password_checkbox_var.get():
        key_entry.config(show="")
    else:
        key_entry.config(show="*")

def submit():
    key_entry.delete(0, END)
    path_label.configure(text="Password Submitted")

# Function to get name from user and store it
def select_image():
    global path

    path = filedialog.askopenfilename()
    path_label.config(text="Image selected: {}".format(path))
    if path:
      encrypt_button = Button(root, text="Encrypt", command=encryptor)
      encrypt_button.pack(side=LEFT, padx=10, pady=10)
      decrypt_button = Button(root, text="Decrypt", command=decryptor)
```

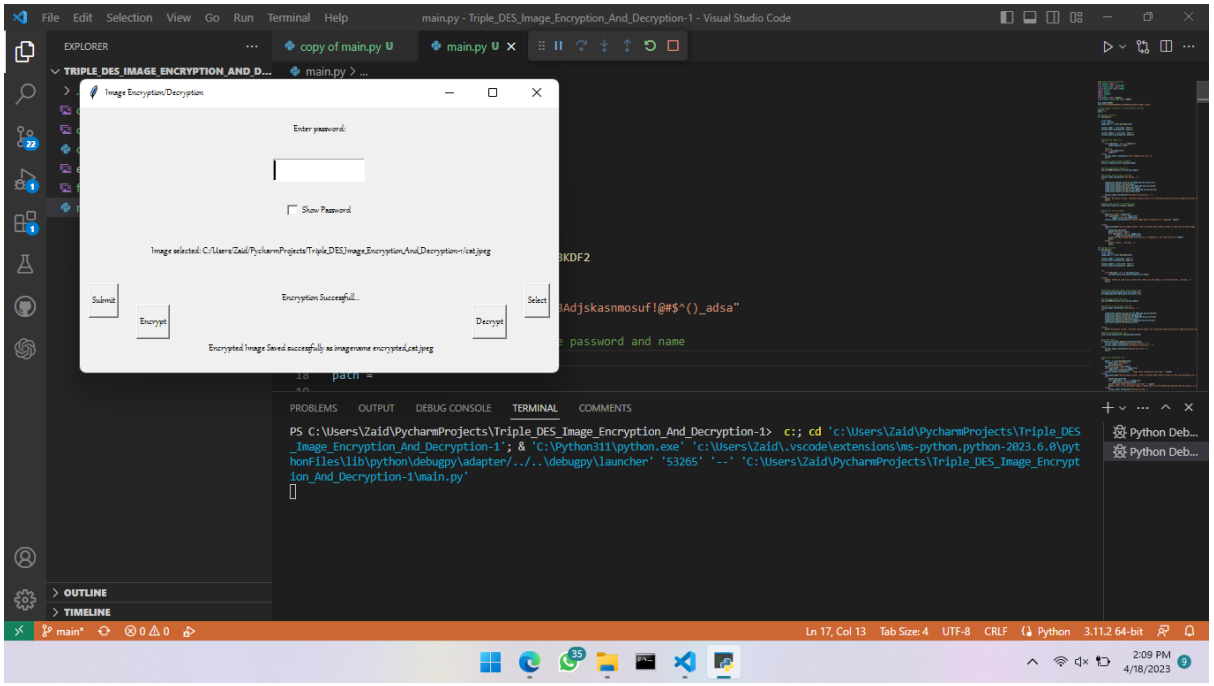
```
      decrypt_button.pack(side=RIGHT, padx=10, pady=10)
    else:
        path_label.config(text="Please Select an Image.")


# Create the main window
root = Tk()
root.title("Image Encryption/Decryption")
root.geometry("300x300")

# Create widgets

key_label = Label(root, text="Enter password:")
key_entry = Entry(root, show="*")
show_password_checkbox_var = IntVar()
show_password_checkbox = Checkbutton(root, text="Show Password",
variable=show_password_checkbox_var, command=get_key)
submit_button = Button(root, text="Submit", command=submit)
path_button = Button(root, text="Select", command=select_image)
path_label = Label(root, text="")


# Position widgets
key_label.pack(padx=10, pady=10)
key_entry.pack(padx=10, pady=10)
show_password_checkbox.pack(padx=10, pady=5)
submit_button.pack(side=LEFT,padx=10, pady=5)
path_button.pack(side=RIGHT,padx=10, pady=10)
path_label.pack(pady=10)


# Start the main loop
root.mainloop()
```

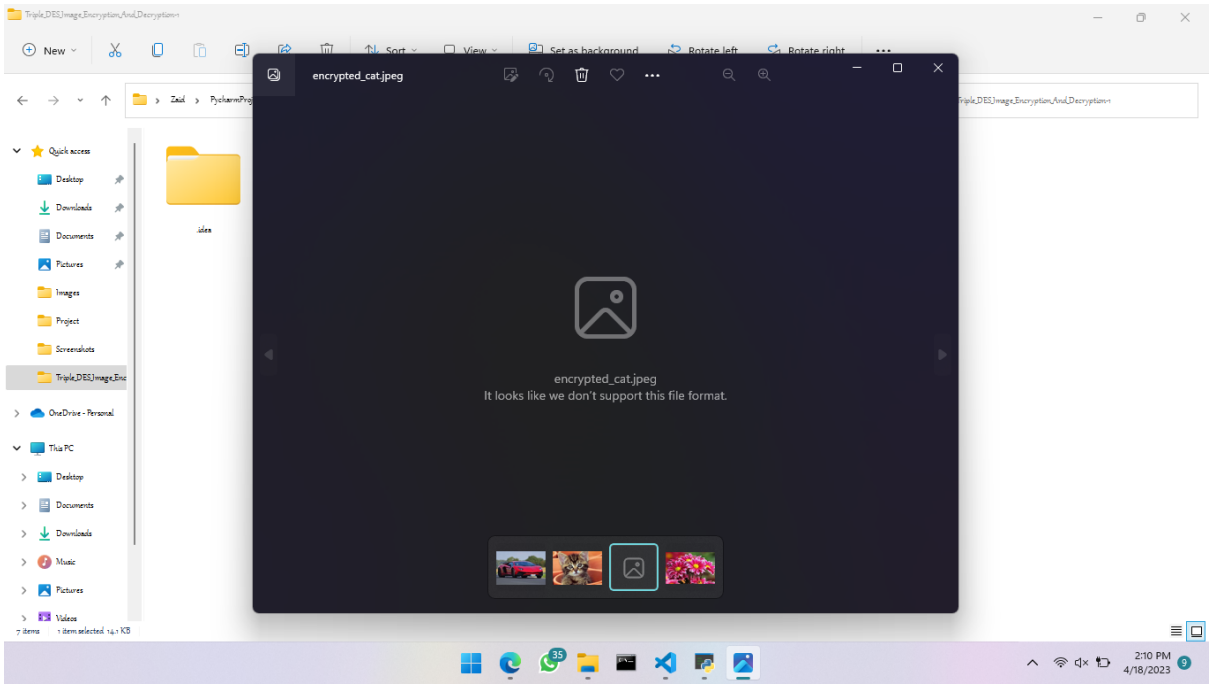
OUTPUT

For Encryption:

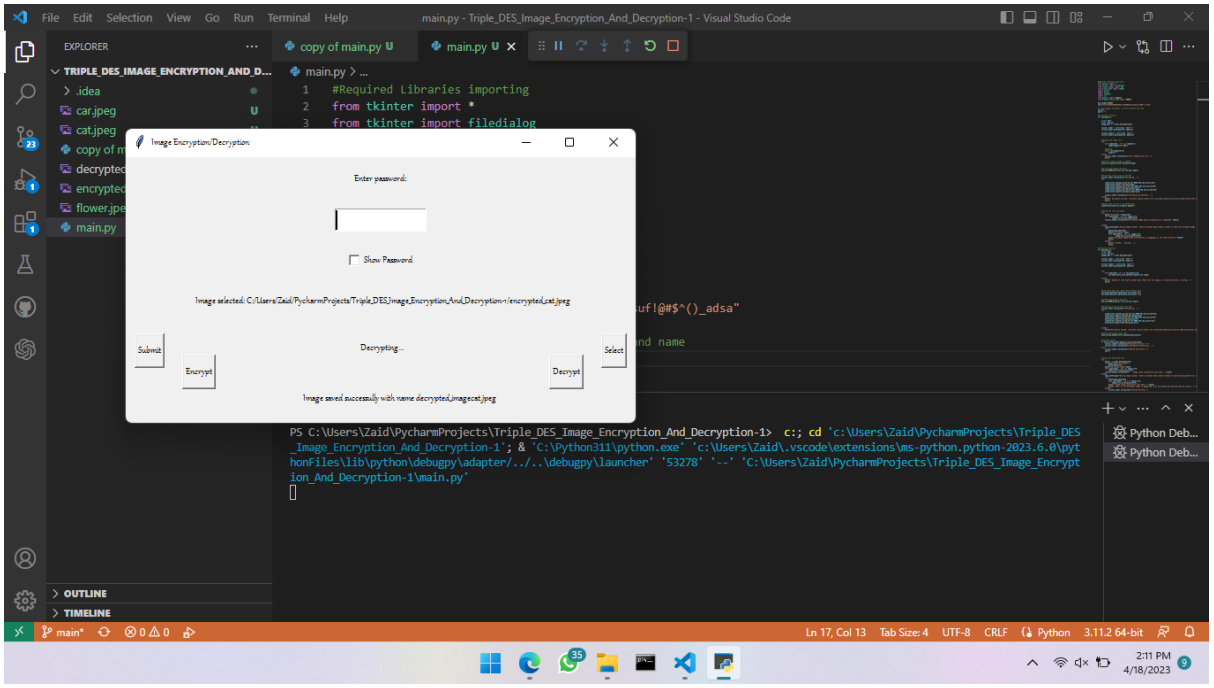





File Edit Selection View Go Run Terminal Help
main.py - Triple_DES_Image_Encryption_And_Decryption-1 - Visual Studio Code
EXPLORER
copy of main.py U
main.py U
TRIPLE_DES_IMAGE_ENCRYPTION_AND_D...
main.py > ...
Image Encryption/Decryption
Enter password:
Show Password
Image selected. C:/Users/Zaid/PycharmProjects/Triple_DES_Image_Encryption_And_Decryption-1/cat.jpeg
Submit
Encrypt
Encryption Successfull...
Decrypt
Select
Encrypted Image Saved successfully as imagename encrypted_cat.jpeg
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
COMMENTS
PS C:\Users\Zaid\PycharmProjects\Triple_DES_Image_Encryption_And_Decryption-1> c:; cd 'c:\Users\Zaid\PycharmProjects\Triple_DES_Image_Encryption_And_Decryption-1'; & 'C:\Python311\python.exe' 'c:\Users\Zaid\.vscode\extensions\ms-python.python-2023.6.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher' '53265' '--' 'C:\Users\Zaid\PycharmProjects\Triple_DES_Image_Encryption_And_Decryption-1\main.py'
Python Deb...
Python Deb...
OUTLINE
TIMELINE
main*
Ln 17, Col 13
Tab Size: 4
UTF-8
CRLF
Python
3.11.2 64-bit
2:09 PM
4/18/2023


New
Sort
View
Set as background
Rotate left
Rotate right
encrypted_cat.jpeg
encrypted_cat.jpeg
It looks like we don't support this file format.
Quick access
Desktop
Downloads
Documents
Pictures
Images
Project
Screenshots
OneDrive - Personal
This PC
Desktop
Documents
Downloads
Music
Pictures
Videos
2:10 PM
4/18/2023

For Decryption:



CHAPTER 8
REFERENCES

1. https://www.tutorialspoint.com/cryptography/data_encrypton_standard.htm
2. https://www.simplelearn.com/what-is-des-article
3. https://pycryptodome.readthedocs.io/en/latest/src/cipher/des3.html
4. https://www.hypr.com/data-encryption-standard-des/
5. https://www.geeksforgeeks.org/data-encryption-standard-des-set-1

CHAPTER 9
CONCLUSION

As the world becomes increasingly digital and interconnected, the need for secure image encryption will only continue to grow. TDES offers a powerful and reliable solution for protecting sensitive and confidential images from unauthorized access and theft.

With ongoing advancements in technology and cryptography, it is likely that TDES will continue to evolve and improve in the years to come, providing even greater levels of security and reliability for image encryption