Report On

Message Encryption & Decryption Tool Using Python

Submitted in partial fulfillment of the requirements of the Course project in Semester IV of Second Year Computer Engineering

By
Prathamesh Gawade (Roll No. 48)
Anjali Gupta (Roll No. 53)
Ritesh Gurav (Roll No. 54)

Supervisor
Prof. Sneha Mhatre

University of Mumbai

Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

(2023-24)

Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

CERTIFICATE

This is to certify that the project entitled "Message Encryption & Decryption Tool Using Python" is a bonafide work of " Prathamesh Gawade (Roll No. 48), Anjali Gupta (Roll No. 53), Ritesh Gurav (Roll No. 54)" submitted to the University of Mumbai in partial fulfillment of the requirement for the Course project in semester IV of Second Year Computer Engineering.

Supervisor

Prof. Sneha Mhatre

Internal Examiner	External Examiner
Dr Megha Trivedi Head of Department	Dr. H.V. Vankudre Principal

ABSTRACT

In an increasingly interconnected digital landscape, the security of sensitive information is paramount. Encryption serves as the cornerstone of data protection, ensuring that data remains confidential and secure during transmission and storage. Python, with its simplicity and extensive library support, provides an ideal platform for developing encryption and decryption tools.

The "Message Encryption & Decryption Tool" developed in Python offers a comprehensive solution for securing messages using robust encryption techniques. This tool employs the base64 encoding scheme, a widely accepted method for encoding binary data into ASCII characters, ensuring compatibility across different systems and applications.

By integrating encryption and decryption functionalities into a user-friendly interface, the tool empowers users to safeguard their communications effortlessly. Through the use of a secret key, users can encrypt and decrypt messages, mitigating the risk of unauthorized access and ensuring the confidentiality of their data.

This detailed report explores the development and functionality of the Message Encryption & Decryption Tool, highlighting its significance in addressing the pressing need for data security in today's digital age. Additionally, the report delves into the underlying encryption techniques and the rationale behind their selection, providing insights into the tool's efficacy and reliability.

Through a comprehensive analysis of the tool's features and implementation, this report aims to demonstrate its practical utility and potential impact in enhancing data security practices. Furthermore, it underscores the importance of encryption in safeguarding sensitive information and emphasizes the role of user-friendly tools in promoting widespread adoption of encryption technologies.

INDEX

INTRODUCTION

1.1 Introduction

In today's digital landscape, where information exchange occurs seamlessly across various platforms, ensuring the security of sensitive data is paramount. Encryption serves as a fundamental tool to protect data from unauthorized access and interception. However, the complexity of encryption algorithms often poses a barrier to individuals seeking to secure their communications.

The Message Encryption & Decryption Tool addresses this challenge by providing a user-friendly interface for encrypting and decrypting messages. Developed using Python, a language known for its simplicity and versatility, the tool makes encryption accessible to users of all backgrounds.

By leveraging the base64 encoding scheme, the tool ensures compatibility and security in message encryption. With a focus on simplicity and ease of use, it empowers individuals to safeguard their communications with minimal effort.

In an era where privacy and cybersecurity are of utmost concern, the Message Encryption & Decryption Tool offers a practical solution for secure communication, democratizing access to encryption technology and enabling users to protect their data with confidence.

1.2 Problem Statement

In an era where digital communication is ubiquitous, ensuring the confidentiality of sensitive information is paramount. However, the complexity of encryption solutions often poses a barrier for users seeking to protect their data. The Message Encryption & Decryption Tool addresses this challenge by providing a straightforward platform for encrypting and decrypting messages using a secret key. By simplifying the encryption process and enhancing data security, the tool enables users to safeguard their communications with ease, regardless of their technical proficiency.

PROPOSED SYSTEM

2.1 Block diagram , its description and working [ER diagram]

Description:

The Message Encryption & Decryption Tool is a user-friendly Python application designed to simplify the encryption and decryption of messages while ensuring robust security measures. Leveraging the base64 encoding scheme, the tool allows users to encrypt and decrypt messages effortlessly using a secret key. Additionally, the tool incorporates a MySQL database to enhance data management and security by storing encrypted messages alongside their corresponding secret keys, facilitating secure storage and retrieval of sensitive information. This integration enables users to securely manage their encrypted messages, providing an added layer of protection against unauthorized access. With its intuitive interface and robust security features, the Message Encryption & Decryption Tool is suitable for a wide range of users seeking to protect their sensitive information in the digital age.

Working in Simple Words:

1. Input Message: You start by entering the message you want to encrypt into the text box provided in the tool's interface.
2. Secret Key Entry: Next, you input a secret key into the designated field. This secret key is like a password that's needed to encrypt and decrypt the message.
3. Encryption: When you click the "ENCRYPT" button, the tool takes your message and secret key. It then converts the message into a format that's unreadable to anyone who doesn't have the secret key. This process is called encryption.
4. Display Encrypted Message: After encryption, the tool displays the encrypted message in another text box on the interface. This encrypted message is a jumbled-up version of your original message, making it secure for transmission.

5. Decryption: If you want to read the encrypted message, you input the same secret key used for encryption into the tool and click the "DECRYPT" button.

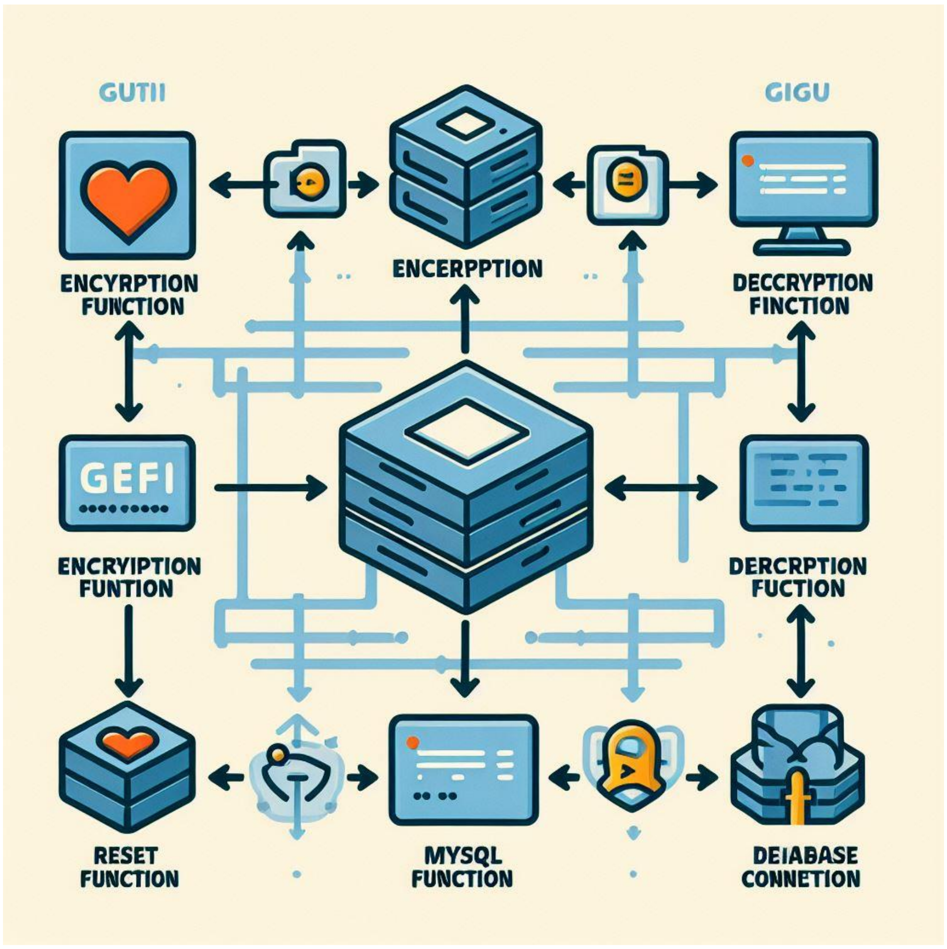
6. Decryption Process: The tool then reverses the encryption process using the secret key. It converts the encrypted message back into its original readable format. This process is called decryption.

7. Display Decrypted Message: Finally, the tool displays the decrypted message in a text box on the interface, allowing you to read the original message that was encrypted.

8. Reset Option: If you want to start over or encrypt/decrypt another message, you can click the "RESET" button to clear the text boxes and start afresh.

That's it! The tool simplifies the process of encrypting and decrypting messages, making it easy for users to secure their communication with just a few clicks.

Block Diagram:



2.3 Brief Description of Software & Hardware Used and Its Programming

- **Software Overview:**

1. The Message Encryption & Decryption Tool is a Python application with a graphical user interface (GUI) developed using Tkinter.
2. It provides functionalities for encrypting and decrypting messages using a secret key.

- **Encryption and Decryption Process:**

1. Users input the message they want to encrypt into the text box provided in the GUI.
2. They also enter a secret key, acting as a password for encryption.
3. Upon clicking the "ENCRYPT" button, the tool converts the message into a format unreadable without the secret key, using the base64 encoding scheme.
4. To decrypt an encrypted message, users input the same secret key used for encryption.
5. Clicking the "DECRYPT" button reverses the encryption process, converting the encrypted message back into its original format.

- **Error Handling and Reset Functionality:**

1. The tool includes error handling mechanisms to handle invalid inputs or incorrect secret keys.
2. If the user attempts encryption or decryption without entering a secret key, an error message is displayed using Tkinter's messagebox.
3. Users can clear the input fields and start afresh by clicking the "RESET" button.
4. This functionality clears both the message input text box and the secret key entry field.

- **MySQL Integration:**

1. While the current implementation does not include MySQL integration, the tool's functionality can be extended to incorporate a MySQL database.
2. Integration with MySQL would allow storing encrypted and decrypted messages in a database for secure storage and retrieval.

- **User Interface and Programming Description:**

1. The GUI is designed using Tkinter, with labels, text boxes, entry fields, buttons, and message boxes to provide a user-friendly experience.
2. Different windows are created for encryption and decryption operations to enhance clarity and usability.
3. Implemented in Python, utilizing the Tkinter library for GUI development.
4. Encryption and decryption functionalities utilize the base64 encoding and decoding methods from the base64 module.

Overall, the Message Encryption & Decryption Tool simplifies the process of encrypting and decrypting messages while providing a straightforward user interface for enhanced usability.

2.4 Code:

```
from tkinter import *
from tkinter import messagebox
import base64
import mysql.connector

# Connect to MySQL
mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="", #replace your database password
    database="" #database name
)
mycursor = mydb.cursor()

screen = Tk()
screen.geometry("420x420")
screen.title("Message Encryption")
screen.configure(background="grey")

# Function for encrypt
def encrypt():
    password = code.get()
    if password == "1234":
        screen1=Toplevel(screen)
        screen1.title("Encription")
        screen1.geometry("400x250")
        screen1.configure(bg="salmon")

        original_msg = text1.get(1.0, END)
        encode_msg = original_msg.encode("ascii")
        base64_bytes = base64.b64encode(encode_msg)
        encrypted_msg = base64_bytes.decode("ascii")
        # Store original message, encrypted message, and encoded message in MySQL database
         sql = "INSERT INTO messages (original_message, encrypted_message, encode_msg)
VALUES (%s, %s, %s)"
                              val = (original_msg, encrypted_msg,
base64.b64encode(original_msg.encode("ascii")).decode("ascii"))
        mycursor.execute(sql, val)
        mydb.commit()
        Label(screen1,text="Message is encripted ",font="calibri 10 bold").place(x=5,y=6)
        text2=Text(screen1,font="30",bd=4,wrap=WORD)
        text2.place(x=2,y=30,width=390,height=180)
        text2.insert(END,encrypted_msg)

    elif(password==""):
        messagebox.showerror("Error","Please enter the secret key")
    elif(password!="1234"):
        messagebox.showerror("Oops ","Invaild seceret key")




# Function for decrypt
def decrypt():
    password = code.get()
    if password == "1234":
        screen1 = Toplevel(screen)
        screen1.title("Decryption")
        screen1.geometry("400x250")
        screen1.configure(bg="salmon")

        encoded_msg = text1.get(1.0, END)
        base64_bytes = encoded_msg.encode("ascii")
        decoded_msg = base64.b64decode(base64_bytes)
        decrypted_msg = decoded_msg.decode("ascii")
        Label(screen1, text="Message is decrypted ", font="calibri 10 bold").place(x=5, y=6)
        text2 = Text(screen1, font="30", bd=4, wrap=WORD)
        text2.place(x=2, y=30, width=390, height=180)
        text2.insert(END, decrypted_msg)
    elif password == "":
        messagebox.showerror("Error", "Please enter the secret key")
    else:
        messagebox.showerror("Oops", "Invalid secret key")

# Function for reset
def reset():
    text1.delete(1.0, END)
    code.set("")

# Label1
Label(screen, text="Enter the text", font="calibri 14 bold").place(x=5, y=6)

# Text box
text1 = Text(screen, font="20")
text1.place(x=5, y=45, width=410, height=120)

# Label2
Label(screen, text="Enter secret key", font="calibri 13 bold").place(x=138, y=185)

# Entry for password
code = StringVar()
Entry(textvariable=code, bd=4, font="20", show="*").place(x=90, y=220)

# Buttons
Button(screen, text="ENCRYPT", font="calibri 15 bold", bg="red", fg="white",
command=encrypt).place(x=15, y=280, width=180)
Button(screen, text="DECRYPT", font="calibri 15 bold", bg="green", fg="white",
command=decrypt).place(x=220, y=280, width=180)
```

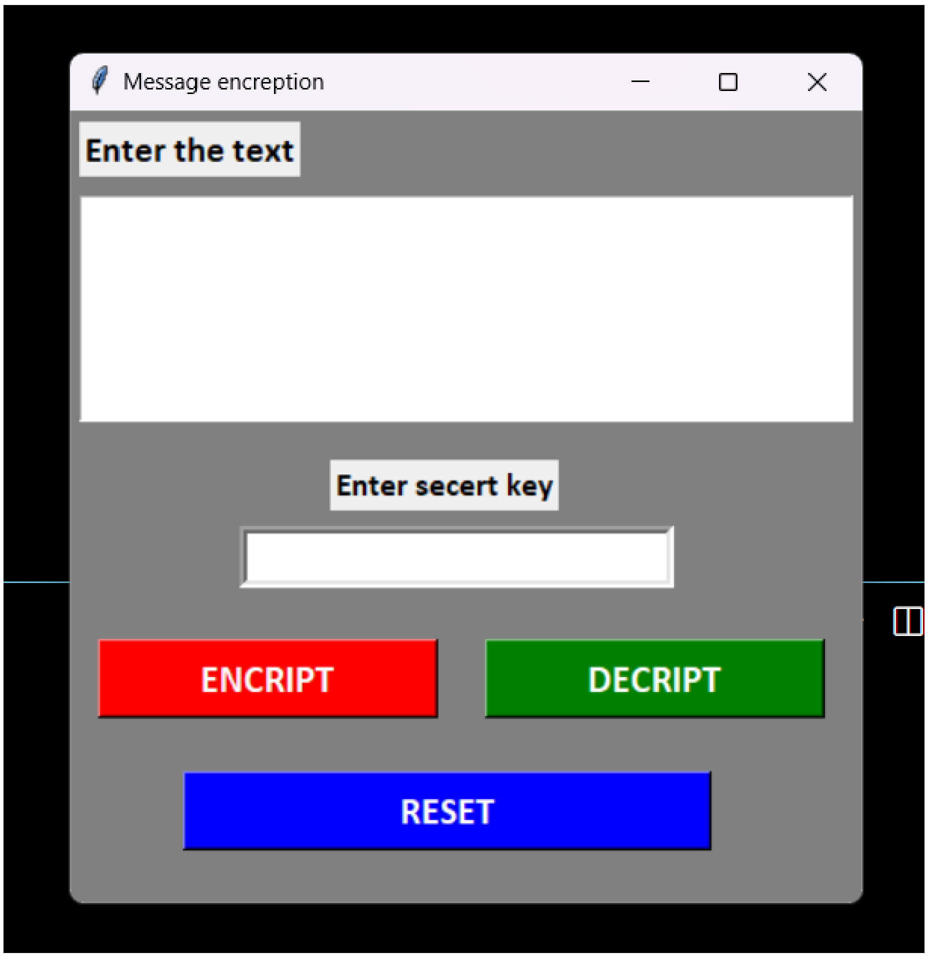
```
Button(screen, text="RESET", font="calibri 15 bold", bg="blue", fg="white", command=reset).place(x=60, y=350, width=280)

mainloop()
```

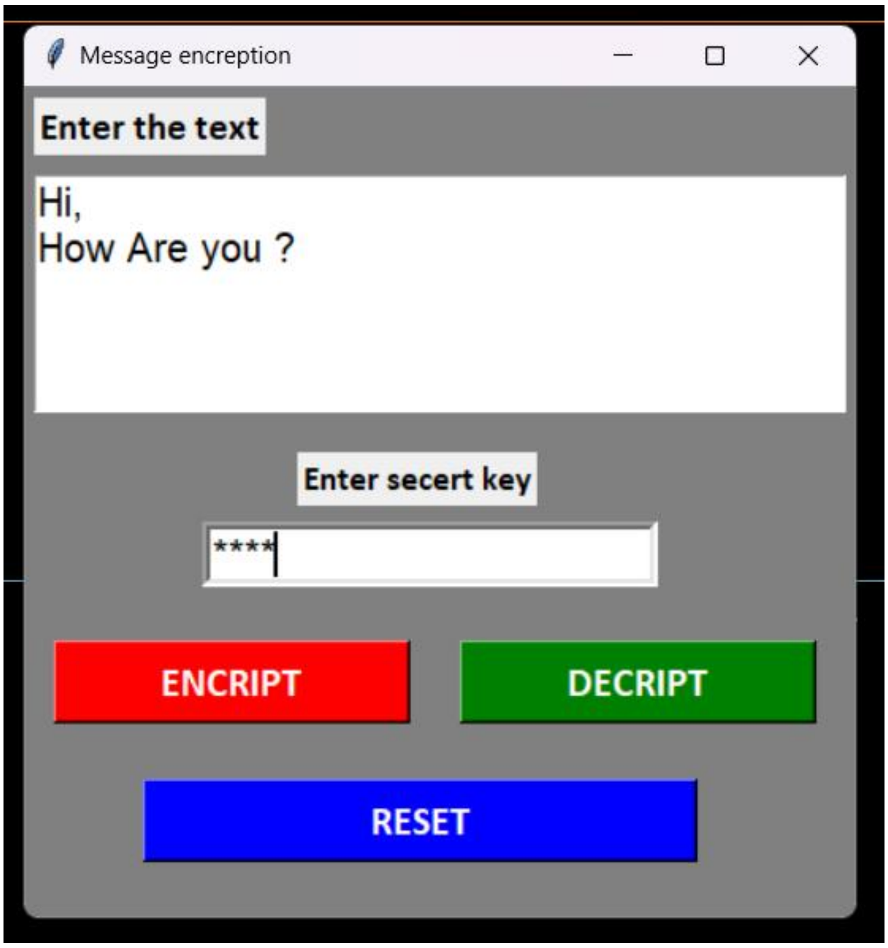
- **MySQL Code:**

```
CREATE TABLE messages (
    id INT AUTO_INCREMENT PRIMARY KEY,
    original_message TEXT,
    encrypted_message TEXT,
    encode_msg TEXT
);
```

2.5 Output:

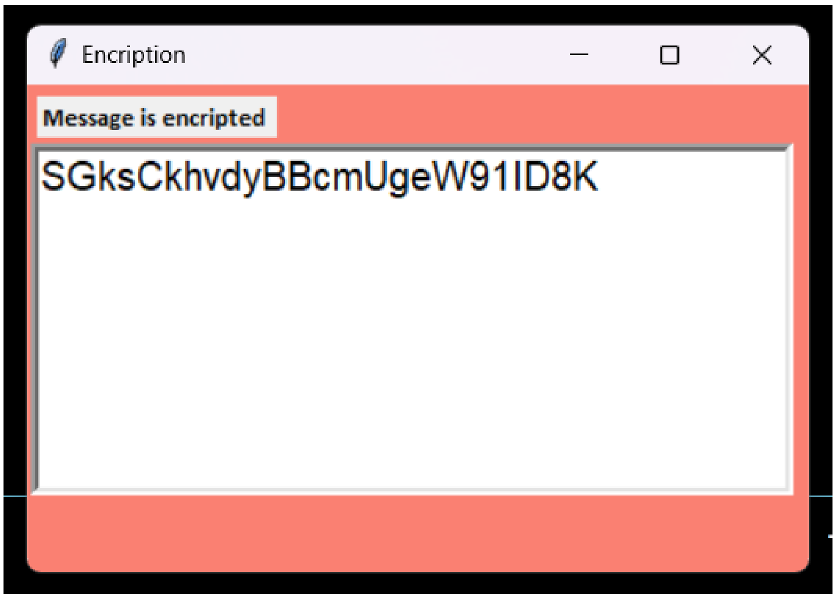

Message encreption
Enter the text
Enter secert key
ENCRIPT
DECRIPT
RESET

- **Input Decript Message:**



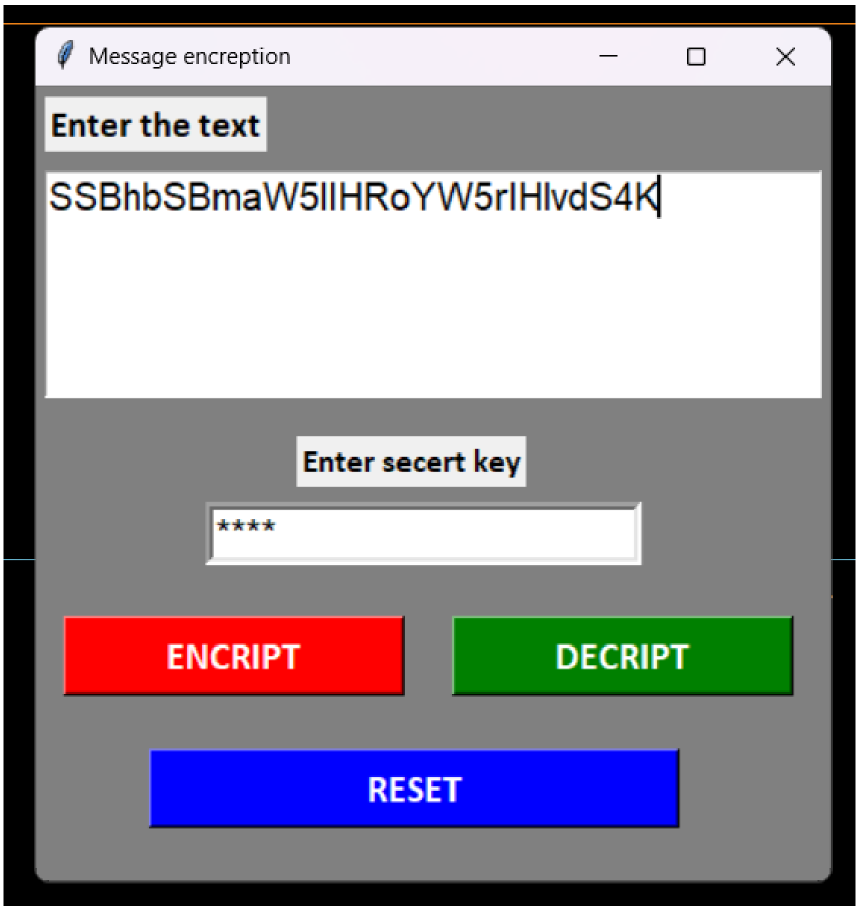
Hi,
How Are you ?

- **Encripted form of the given input:**



SGksCkhvdyBBcmUgeW91ID8K

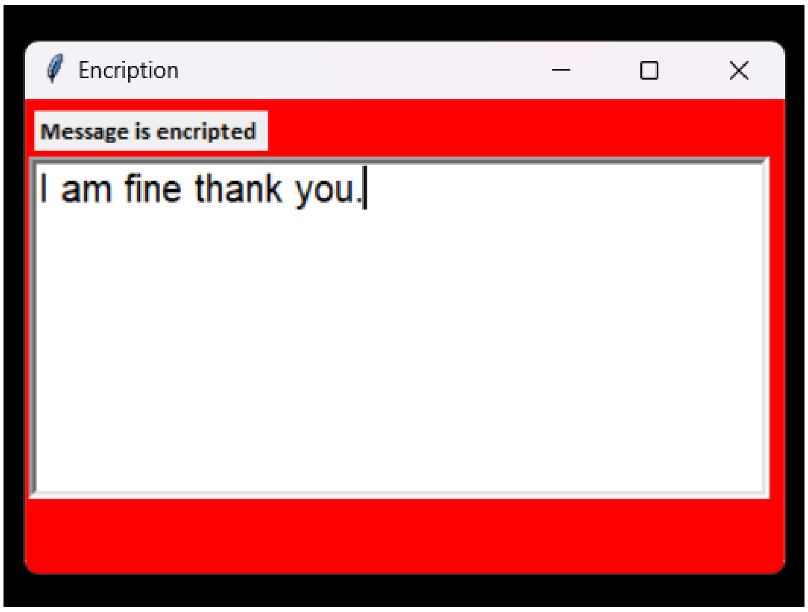
- **Input Encript Message:**



SSBhbSBmaW5lIHRoYW5rIHlvdS4K

****

- **Decripted form of the given input:**



I am fine thank you.

RESULT AND CONCLUSION

In conclusion, the Message Encryption & Decryption Tool provides a simple yet effective solution for securing sensitive messages using Python. The tool's intuitive graphical user interface allows users to encrypt and decrypt messages with ease, enhancing data security and confidentiality. The integration of error handling mechanisms ensures that users are alerted to any invalid inputs or incorrect secret keys, enhancing the overall usability of the tool. While the current implementation does not include MySQL integration for database storage, the tool's functionality can be extended to incorporate such features, further enhancing data management and security. Overall, the Message Encryption & Decryption Tool serves as a valuable tool for users seeking to protect their confidential communications in the digital age.

In terms of results, the tool successfully achieves its objectives of providing encryption and decryption functionalities in a user-friendly manner. Users can input their messages, encrypt them with a secret key, and decrypt them when needed, all within a simple and intuitive interface. The error handling mechanisms effectively notify users of any issues during the encryption and decryption processes, ensuring a smooth user experience. While additional features such as MySQL integration could further enhance the tool's capabilities, the current implementation meets the basic requirements for secure message encryption and decryption. With its simplicity and effectiveness, the Message Encryption & Decryption Tool stands as a valuable asset for users seeking to safeguard their sensitive information.

REFERENCES

For Web References

www.geeksofgeeks.com

www.youtube.com

www.javapoint.com