Secure sensitive information by encoding and decoding messages

A project report submitted to
MALLA REDDY UNIVERSITY
in partial fulfillment of the requirements for the award of degree of

BACHELOR OF TECHNOLGY
in
COMPUTER SCIENCE & ENGINEERING (AI & ML)

Submitted by

V. JYOSHITA REDDY	**:**	**2111CS020193**
R.JYOSHNA	**:**	**2111CS020194**
B.JYOTHI	**:**	**2111CS020195**
RITHIKA.J.POOJARI	**:**	**2111CS020196**
N.KARTHEEK	**:**	**2111CS020197**
B.KARTHIK GOUD	**:**	**2111CS020198**

Under the Guidance of

PROF.N.V.P.R.RAJESWARI

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING (AI & ML)

MALLA REDDY UNIVERSITY
(Telangana State Private Universities Act No.13 of 2020 and G.O.Ms.No.14, Higher Education (UE) Department)

2023

MALLA REDDY UNIVERSITY

(Telangana State Private Universities Act No.13 of 2020 and G.O.Ms.No.14, Higher Education (UE) Department)

COLLEGE CERTIFICATE

This is to certify that this is the bonafide record of the application development entitled, "**Secure sensitive information by encoding and decoding messages"** Submitted by

V. JYOSHITA REDDY	:	2111CS020193
R.JYOSHNA	:	2111CS020194
B.JYOTHI	:	2111CS020195
RITHIKA.J.POOJARI	:	2111CS020196
N.KARTHEEK	:	2111CS020197
B.KARTHIK GOUD	:	2111CS020198

B.Tech II year II semester, Department of CSE (AI&ML) during the year 2022-23. The results embodied in the report have not been submitted to any other university or institute for the award of any degree or diploma.

PROJECT GUIDE

Prof. N. V. P. R. Rajeswari

HEAD OF THE DEPARTMENT

Dr. Thayyaba Khatoon

CSE(AI&ML)

ACKNOWLEDGEMENT

An endeavor over a long period can be advice and support of many well wishers. We take this opportunity to express our gratitude and appreciation to all of them.

We owe our tribute to **Dr. Thayyaba Khatoon(Head of Department)** , for giving all of us such a wonderful opportunity to explore ourselves and the outside world to work on the real-life scenarios where the machine learning is being used nowadays.

We are very grateful to our project guide **Prof. N. V. P. R. Rajeswari** , for the guidance ,inspiration and constructive suggestions that helped us in the development of this application.

We wish to express our sincere thanks and gratitude to our project mentor **Prof . Sabyasachi** , for the simulating discussions, in analyzing problems associated with our project work and for mentoring us throughout the project.

We are much obliged to **Prof. A .Shiva Kumar(Application Development Coordinator)** for encouraging and supporting us immensely by giving many useful inputs with respect to the topic chosen by us, throughout the development of the application ensuring that our project is a success.

We also thank our parents and family at large for their moral and financial support in funding the project to ensure successful completion of the project .

CONTENTS

CHAPTER 1

INTRODUCTION

In this digital era, the need for security is increasing rapidly. Complying with this requirement, the encryption and decryption algorithms were devised. Encoding and decoding messages can be an effective way to secure sensitive information. There are many methods of encoding and decoding messages, some of which are more secure than others. Digital communication witnesses a noticeable and continuous development in many applications in the Internet. Hence, secure communication sessions must be provided. The security of data transmitted across a global network has turned into a key factor on the network performance measures. So, the confidentiality and the integrity of data are needed to prevent eavesdroppers from accessing and using transmitted data.

OBJECTIVE OF PROJECT

This objective of this project is to encode and decode messages using a common key. In this project, users have to enter the message to encode or decode. Users have to select the mode to choose the encoding and decoding process. The same key must be used to process the encoding and decoding for the same message.

1.1ABSTRACT

The project aim to develop a secure sensitive information by encoding messages. Message encoding and decoding is the process to first convert the original text to the random and meaningless text called ciphertext. This process is called encoding. Decoding is the process to convert that ciphertext to the original text. This process is also called the Encryption, Decryption process. This objective of this project is to encode and decode messages using a common key. This project will be built using the Tkinter and base64 library .In this project, users have to enter the message to encode or decode. Users have to select the mode to choose the encoding and decoding process. The same key must be used to process the encoding and decoding for the same message. Base64 is a library that allows the user to encode and decode the string. The string to be encoded should be in byte form. A function to encode binary information to ASCII characters then decode those ASCII classic characters to binary data is provided by the base64 module of the standard GUI Python library, Tkinter. On the command prompt, we use the pip install command to install the library. In addition to base64, base 85 is also used to provide additional encoding. Base64 and Base84 are included in thenewer versions of Python3, so be sure to check the version beforehand.

1.2. Limitations of project

1. It is typically presented as a linear, one-way process.

2. It does not deal adequately with the processing of meaning, it mainly focuses on technological communication

3. Insufficient learning that deeper neural networks tend to encounter the notorious problem of vanishing gradients.

4. Asymmetric power relationships will likely lead to different interpretations of message

CHAPTER 2

2. ANALYSIS

The analysis of Stuart Halls's Influential essay offers a densely theoretical account of how messages are produced and disseminated, referring particularly to television. This means that the coding of a message does not control its reception but not transparently each stage has its own determining limits and possibilities. Encoding and decoding messages can be an effective way to secure sensitive information. There are many methods of encoding and decoding messages, some of which are more secure than others. One simple method is the Caesar cipher, which involves shifting each letter of the message by a certain number of positions in the alphabet. For example, if the shift is three, the letter A would become D, B would become E, and so on. To decode the message, the recipient would simply shift each letter back by the same number of positions. However, this method is not very secure, as it is easily broken by frequency analysis, where an attacker analyzes the frequency of each letter in the encoded message and compares it to the frequency of letters in the English language. More advanced encryption methods, such as the Advanced Encryption Standard (AES), use complex algorithms to encrypt data that are much more secure. It's important to note that encoding and decoding messages is just one part of a larger data security strategy. Other important steps include using secure communication channels, limiting access to sensitive information, and regularly updating security protocols to address new threats.

2.1 Software requirement specification

2.1.1 Software requirements

- OS: windows 11
- Python
- Tkinter
- Jupyter Notebook
- Libraries like Numpy, pandas.

2.1.2. Hardware requirements

- Processor: INTEL.
- RAM: Minimum of 256MB or higher.
- HOD: 10GB or higher.
- Monitor:15” or 17” color monitor.
- Keyboard: Standard 110 keys keyboard.

2.2 Existing System

Digital encryption algorithms work by manipulating the digital content of a plaintext message mathematically, using an encryption algorithm and a digital key to produce a cipher text version of the message. One of the significant disadvantages of encryption is key management. Key management should be done efficiently as Encryption and decryption keys cannot be compromised, which might invalidate the data security measures taken.

2.3 Proposed System

In this proposed method first, the message is encrypted by use RSA algorithm. There will be an agreement between the sender and the receiver about the key for the concealment algorithm as well as the key for the encryption algorithm or these keys may be exchanged by a secure communication method. Our method starts by encryption first then hidden crypted data. Before applying the modules we convert input into Base-64 and we save the obtained text in a text file. where as encrypted data is called Ciphertext, and decrypted data is called plain text.

2.4 Modules

- Importing the base64 and tkinter modules.
- Creating the GUI using tkinter.
- Creating the functions and the button.

2.5 Architecture

After designing the working principle, the flow chart of the system is implemented where the code and the model is developed and tested. The flowchart of the complete system in shown in Fig 2.5.1

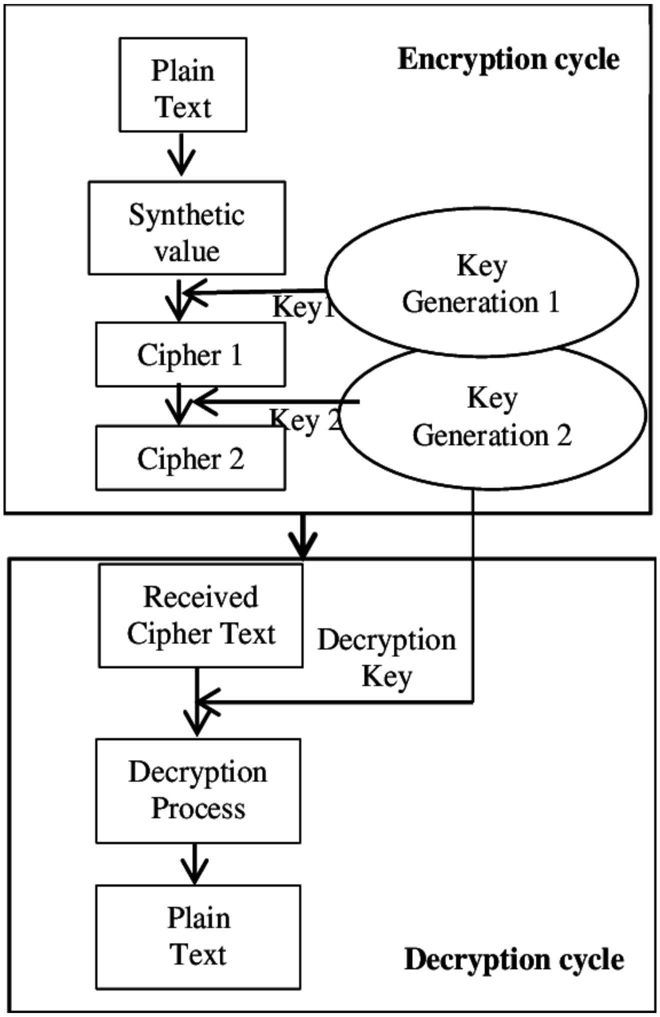


Fig:2.5.1

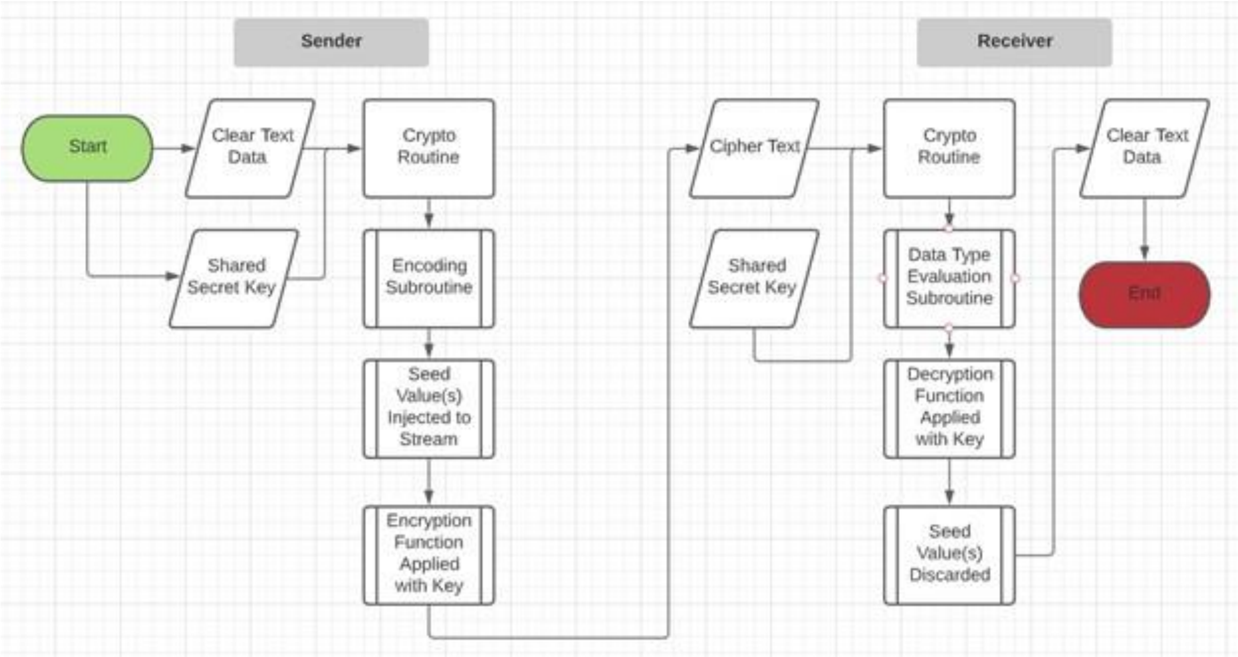

Sender
Receiver
Start
Clear Text Data
Shared Secret Key
Crypto Routine
Encoding Subroutine
Seed Value(s) Injected to Stream
Encryption Function Applied with Key
Cipher Text
Shared Secret Key
Crypto Routine
Data Type Evaluation Subroutine
Decryption Function Applied with Key
Seed Value(s) Discarded
Clear Text Data
End

Fig: 2.5.2

CHAPTER 3

3. DESIGN

Project design is a major step towards a successful project. A project design is a strategic organization of ideas, materials and processes for the purpose of achieving a goal. Project managers rely on a good design to avoid pitfalls and provide parameters to maintain crucial aspects of the project. Project design is an early phase of the project where a project's key features, structure, criteria for success, and major deliverables are all planned out. The point is to develop one or more designs which can be used to achieve the desired project goals. Stake holders can then choose the best design to use for the actual execution of the project. So, the design can be implemented using Unified Modeling Language. diagrams such as class diagram, use case diagram, sequence diagram, activity diagrams. UML offers away to visualize a system's architectural blue prints in a diagram, including elements such as:

UML is a common language for business analysts, software architects and developers used to describe, specify, design, and document existing or new business processes, structure and behavior of artifacts of software systems. The key to making a UML diagram is connecting shapes that represent an objector class with other shapes to illustrate relation ships and the flow of information and data.

- Any activities
- Individual components of the system
- How the system will run

3.1 Class Diagram

A class diagram in the Unified Modeling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object- oriented languages.

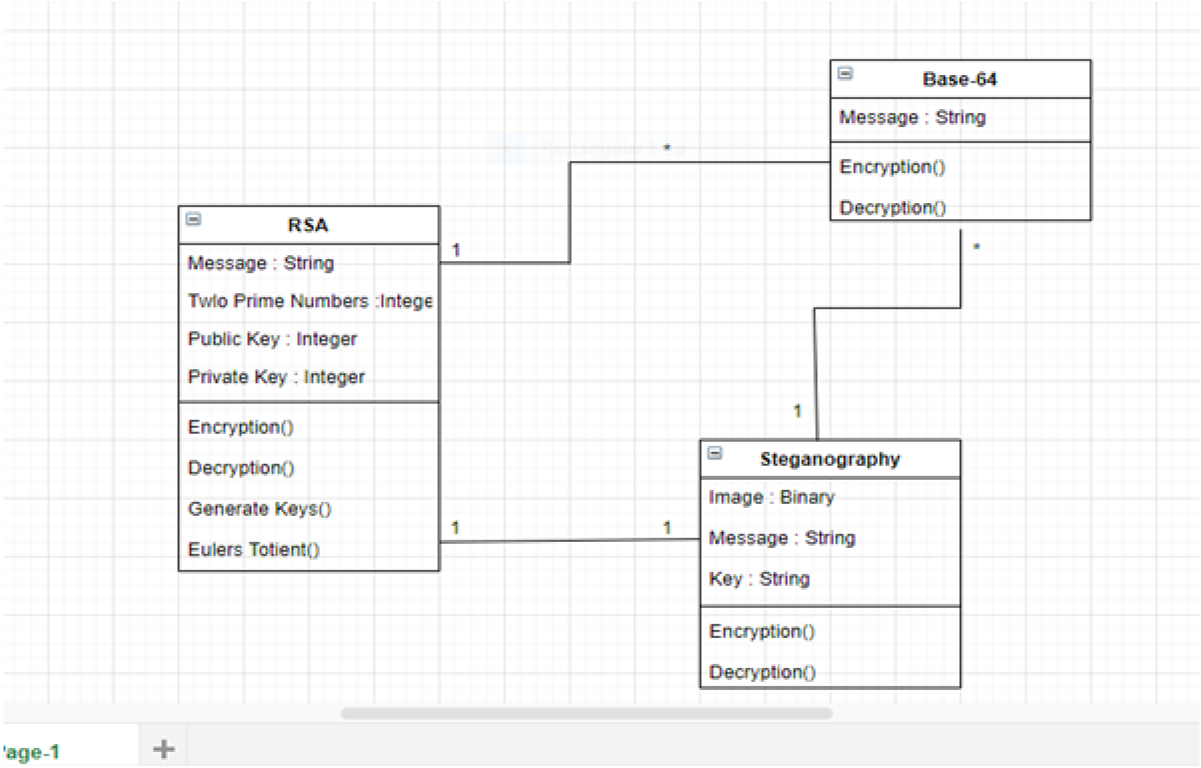


Fig3.1 Class Diagram

3.2. Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.

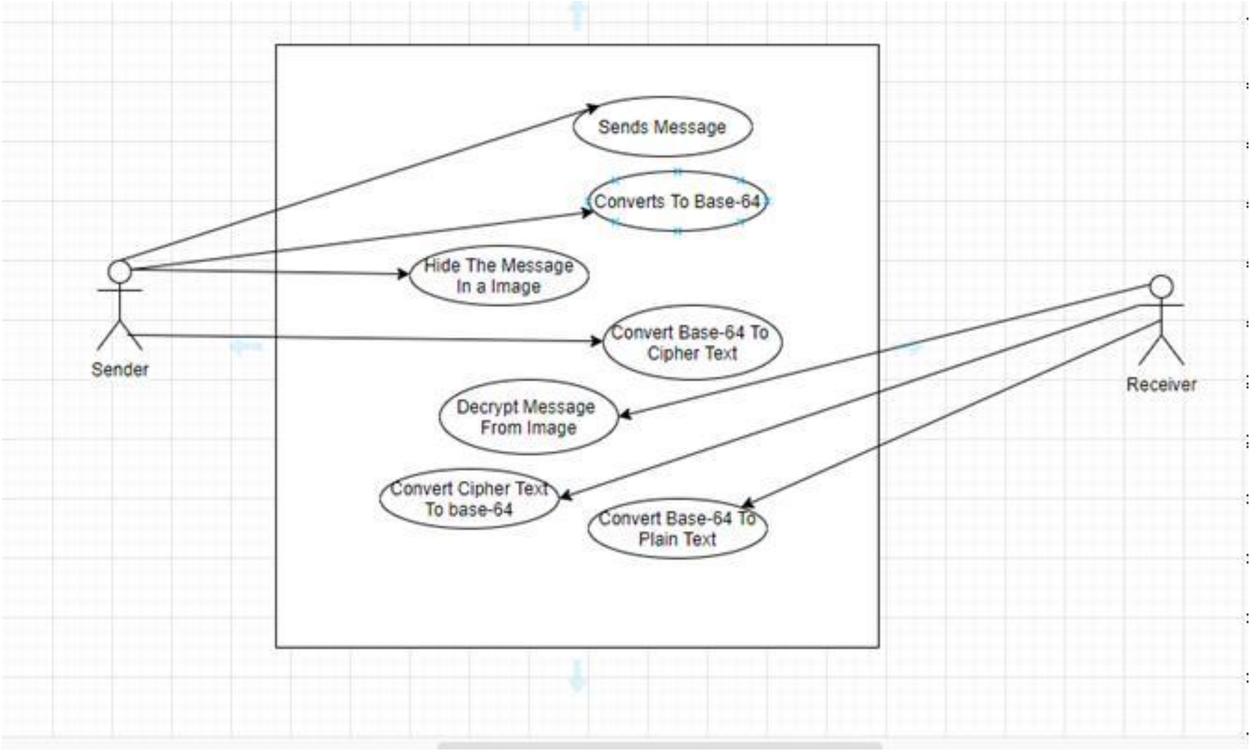

Sends Message
Converts To Base-64
Hide The Message In a Image
Convert Base-64 To Cipher Text
Sender
Decrypt Message From Image
Convert Cipher Text To base-64
Convert Base-64 To Plain Text
Receiver

Fig3.2Use Case Diagram

3.3. Activity Diagram

Activity diagrams are graphical representations of work flows of step wise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows) as well as the data flows intersecting with the related activities. Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores.

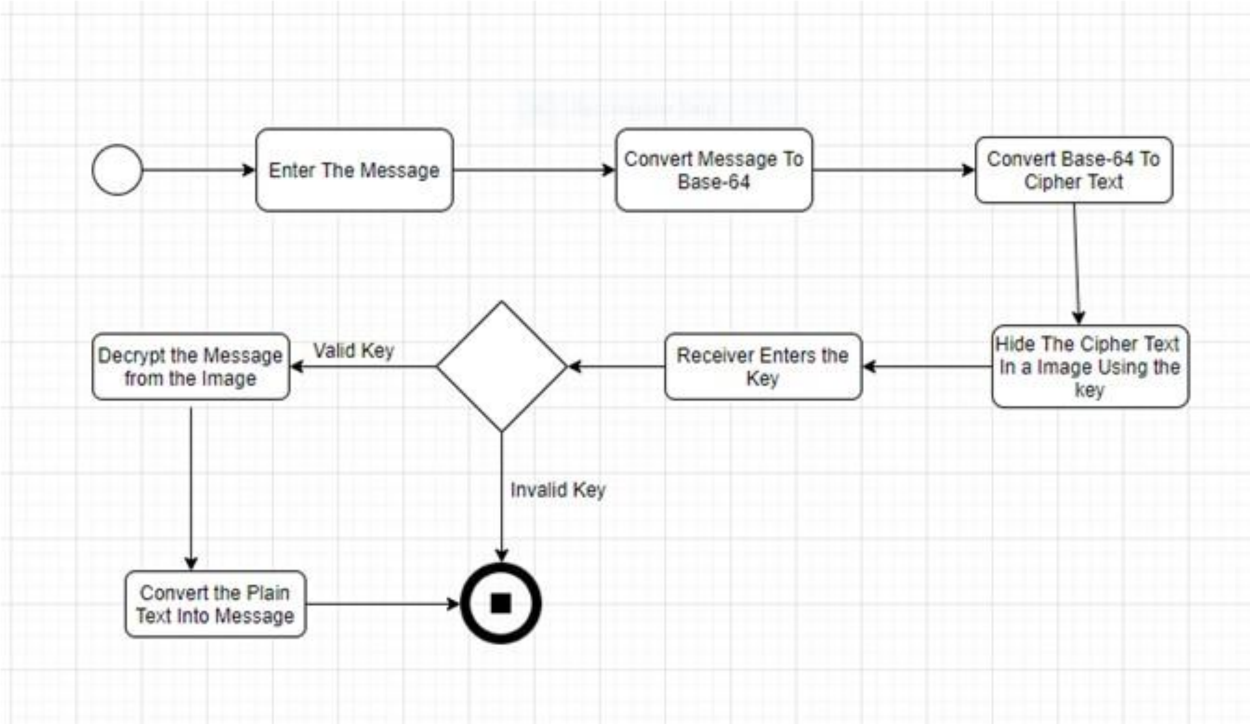

Enter The Message
Convert Message To Base-64
Convert Base-64 To Cipher Text
Hide The Cipher Text In a Image Using the key
Receiver Enters the Key
Valid Key
Decrypt the Message from the Image
Invalid Key
Convert the Plain Text Into Message

Fig3.3 Activity Diagram

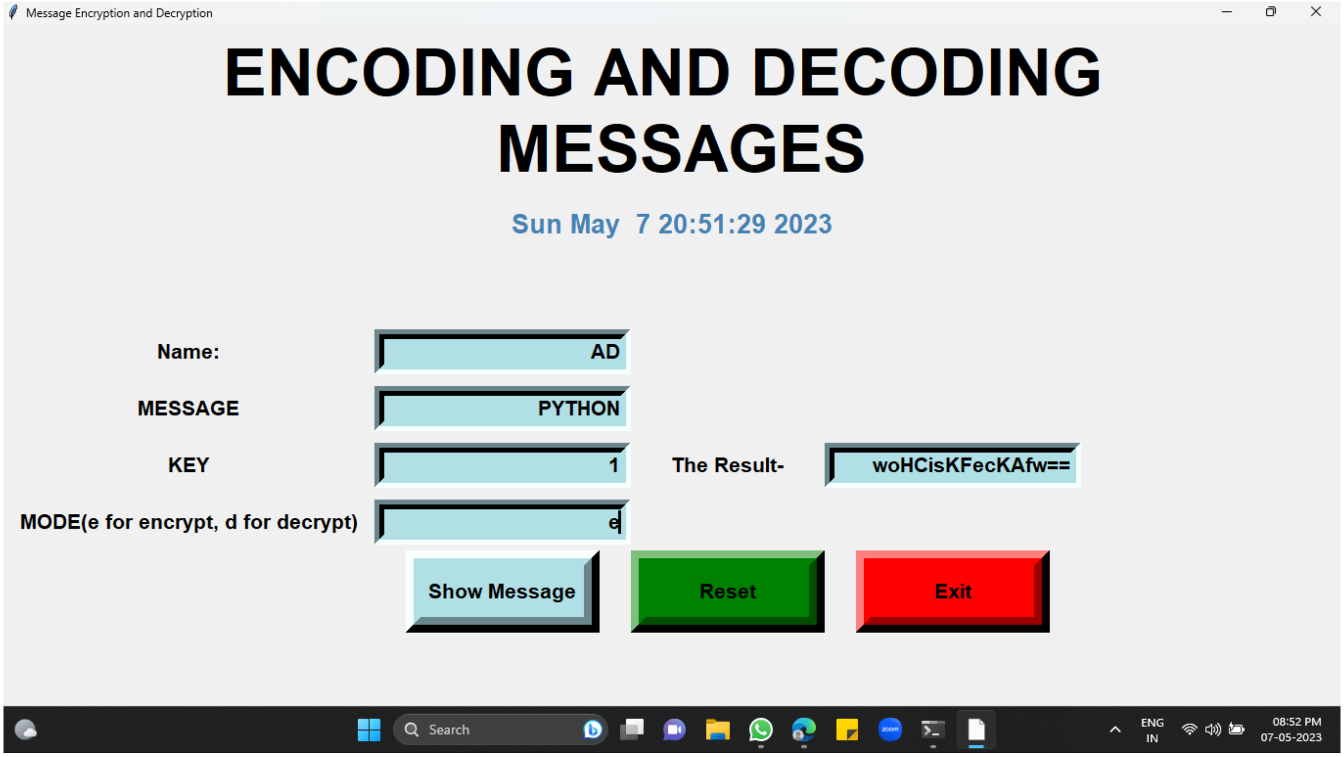
CHAPTER 4

4.DEPLOYMENT AND RESULTS

INTRODUCTION

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter outputs the fastest and easiest way to create the GUI applications. Python provides the Tkinter toolkit to develop GUI applications. Now, it’s upto the imagination or necessity of developer, what he/she want to develop using this toolkit. Let’s try to implement a message encryption-decryption application according to the Vigenère cipher, which can encrypt the message using the key and can decrypt the encrypted hash using same key.

4.1 Output Screens



Message Encryption and Decryption

ENCODING AND DECODING MESSAGES

Sun May 7 20:51:29 2023

Name: AD

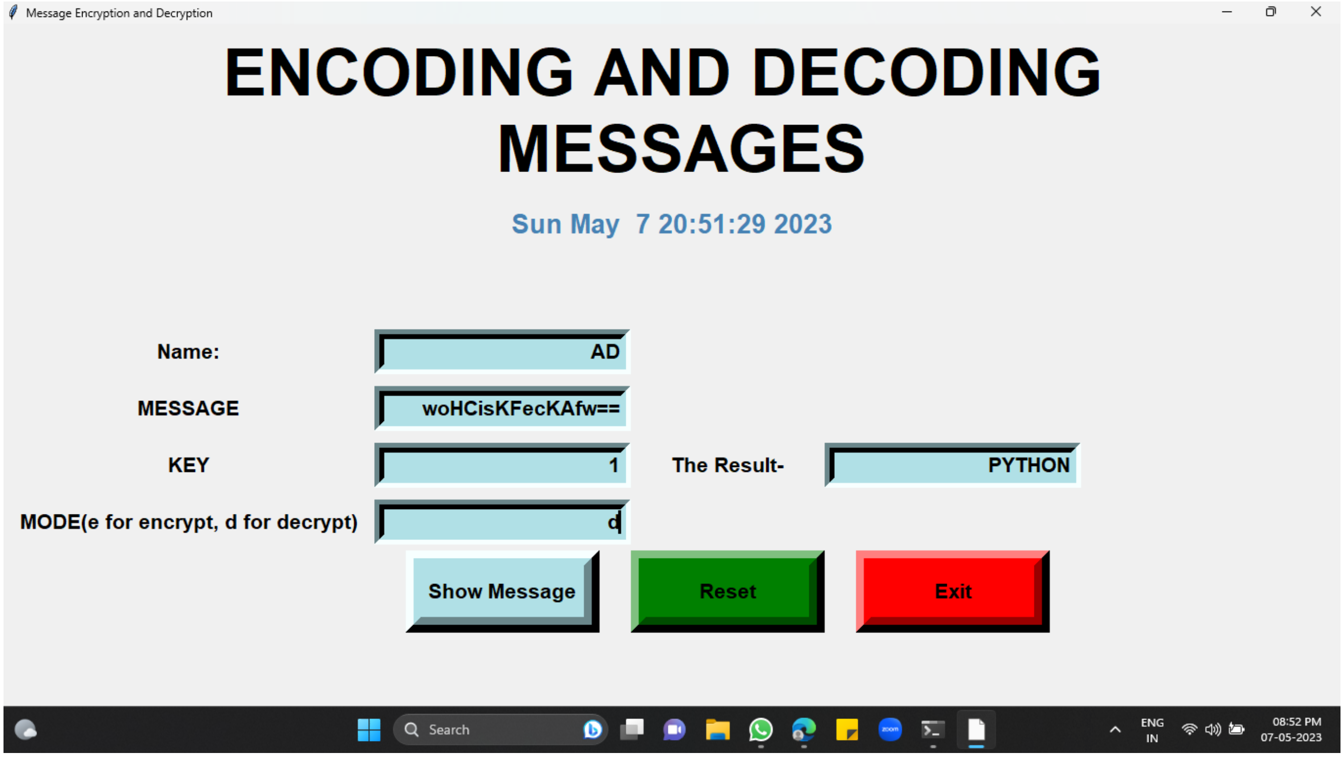
MESSAGE PYTHON

KEY 1

The Result- woHCisKFecKAfw==

MODE(e for encrypt, d for decrypt) e

Fig4.1.1 Encrypt Window



ENCODING AND DECODING MESSAGES

Sun May 7 20:51:29 2023

Name: AD

MESSAGE woHCisKFecKAfw==

KEY 1

The Result- PYTHON

MODE(e for encrypt, d for decrypt) d

Fig4.1.2 Decrypt Window

4.2 source code

Modules used in the project :

Tkinter

GUI tool kit time datetime

base64 -> Vigenère cipher

`Creating the program:`

# import tkinter module

from tkinter import *

# import other necessary modules

import random

import time

import datetim

```
# creating root object
root = Tk()
# defining size of window
root.geometry("1200x6000")
# setting up the title of window
root.title("Message Encryption and Decryption")
Tops = Frame(root, width = 1600, relief = SUNKEN)
Tops.pack(side = TOP)
f1 = Frame(root, width = 800, height = 700,relief = SUNKEN)
f1.pack(side = LEFT)
# ==================================================
#                                    TIME
# ==================================================
localtime = time.asctime(time.localtime(time.time()))
lblInfo = Label(Tops, font = ('helvetica', 50, 'bold'),
                  text = "SECRET MESSAGING \n Vigenère cipher",
                                             fg = "Black", bd = 10, anchor='w')
lblInfo.grid(row = 0, column = 0)
lblInfo = Label(Tops, font=('arial', 20, 'bold'),
                           text = localtime, fg = "Steel Blue",
                                                      bd = 10, anchor = 'w')
lblInfo.grid(row = 1, column = 0)
rand = StringVar()
Msg = StringVar()
key = StringVar()
mode = StringVar()
Result = StringVar()
# exit function
def qExit():
        root.destroy()
# Function to reset the window
```

```
def Reset():
        rand.set("")
        Msg.set("")
        key.set("")
        mode.set("")
        Result.set("")
# reference
lblReference = Label(f1, font = ('arial', 16, 'bold'),
                                text = "Name:", bd = 16, anchor = "w")
lblReference.grid(row = 0, column = 0)
txtReference = Entry(f1, font = ('arial', 16, 'bold'),
                        textvariable = rand, bd = 10, insertwidth = 4,
                                        bg = "powder blue", justify = 'right')
txtReference.grid(row = 0, column = 1)
# labels
lblMsg = Label(f1, font = ('arial', 16, 'bold'),
                text = "MESSAGE", bd = 16, anchor = "w")
lblMsg.grid(row = 1, column = 0)
txtMsg = Entry(f1, font = ('arial', 16, 'bold'),
                textvariable = Msg, bd = 10, insertwidth = 4,
                                bg = "powder blue", justify = 'right')
txtMsg.grid(row = 1, column = 1)
lblkey = Label(f1, font = ('arial', 16, 'bold'),
                        text = "KEY", bd = 16, anchor = "w")
lblkey.grid(row = 2, column = 0)
txtkey = Entry(f1, font = ('arial', 16, 'bold'),
                textvariable = key, bd = 10, insertwidth = 4,
                                bg = "powder blue", justify = 'right')
txtkey.grid(row = 2, column = 1)
lblmode = Label(f1, font = ('arial', 16, 'bold'),
                text = "MODE(e for encrypt, d for decrypt)",bd = 16, anchor = "w")
```

```
lblmode.grid(row = 3, column = 0)
txtmode = Entry(f1, font = ('arial', 16, 'bold'),
                    textvariable = mode, bd = 10, insertwidth = 4,bg = "powder blue", justify = 'right')
txtmode.grid(row = 3, column = 1)
lblService = Label(f1, font = ('arial', 16, 'bold'),text = "The Result-", bd = 16, anchor = "w")
lblService.grid(row = 2, column = 2)
txtService = Entry(f1, font = ('arial', 16, 'bold'),
                              textvariable = Result, bd = 10, insertwidth = 4,
                                                bg = "powder blue", justify = 'right')
txtService.grid(row = 2, column = 3)
# Vigenère cipher
import base64
# Function to encode
def encode(key, clear):
          enc = []
          for i in range(len(clear)):
                    key_c = key[i % len(key)]
                    enc_c = chr((ord(clear[i]) +ord(key_c)) % 256)
                    enc.append(enc_c)
          return base64.urlsafe_b64encode("".join(enc).encode()).decode()
# Function to decode
def decode(key, enc):
          dec = []
          enc = base64.urlsafe_b64decode(enc).decode()
          for i in range(len(enc)):
                    key_c = key[i % len(key)]
                    dec_c = chr((256 + ord(enc[i]) - ord(key_c)) % 256)
                    dec.append(dec_c)
          return "".join(dec)
def Ref():
          print("Message= ", (Msg.get()))
```

```
    clear = Msg.get()
    k = key.get()
    m = mode.get()
    if (m == 'e'):
        Result.set(encode(k, clear))
    else:
        Result.set(decode(k, clear))
# Show message button
btnTotal = Button(f1, padx = 16, pady = 8, bd = 16, fg = "black",
                        font = ('arial', 16, 'bold'), width = 10,
                    text = "Show Message", bg = "powder blue",
                        command = Ref).grid(row = 7, column = 1)
# Reset button
btnReset = Button(f1, padx = 16, pady = 8, bd = 16,
                fg = "black", font = ('arial', 16, 'bold'),
                    width = 10, text = "Reset", bg = "green",
                command = Reset).grid(row = 7, column = 2)
# Exit button
btnExit = Button(f1, padx = 16, pady = 8, bd = 16,
                fg = "black", font = ('arial', 16, 'bold'),
                    width = 10, text = "Exit", bg = "red",
                command = qExit).grid(row = 7, column = 3)
# keeps window alive
root.mainloop()
```

CHAPTER 5

5.1 Conclusion

We have successfully developed Message encode – decode project in Python. We used the popular tkinter library for rendering graphics on a display window and base64 to encode & decode. We learned how to encode and decode the string, how to create button, widget, and pass the function to the button. In this way, we can encode our message and decode the encoded message in a secure way by using the key.

5.2 Future enhancement

The field of encoding and decoding messages is constantly evolving, and there are many potential future enhancements that could improve the security, efficiency, and reliability of these processes. Here are a few examples:

- Quantum Cryptography
- Homomorphic Encryption:
- Machine Learning-Based Encryption
- Blockchain-Based Encryption
- Post-Quantum Cryptography

These are just a few examples of the many potential enhancements to encoding and decoding messages that could emerge in the future. As technology continues to evolve, we can expect to see new and innovative approaches to encryption and communication security.