**A MINI PROJECT REPORT ON**

**Encryption and Decryption of Images**

**Submitted in partial fulfilment for the award of the degree of**

**BACHELOR OF TECHNOLOGY**

**In**

# Electronics and Communication Technology

**By**

## D. Saritha (20A81A1405)

## K. Mohini Durga Bhavani (20A81A1414)

M. Venkata Sai (20A81A1430)

J. Pravallika (20A81A1410)

E. Suraj (20A81A1406)

**Under the Esteemed Supervision of**

**Mr. M Subbarao**

Assistant professor, Department of ECE



**Department of Electronics and Communication Technology**

SRI VASAVI ENGINEERING COLLEGE (Autonomous) (Affiliated to JNTUK, Kakinada) Pedatadepalli, Tadepalligudem-534101, A.P

2021-2024

## SRI VASAVI ENGINEERING COLLEGE (Autonomous) Department of Electronics and Communication Engineering



**Pedatadepalli, Tadepalligudem**



## This is to certify that the Project Report entitled “ENCRYPTION AND DECRYPTION OF IMAGES” submitted by D. Saritha(20A81A1405), K. Mohini Durga Bhavani(20A81A1414), M. Venkata Sai(20A81A1430), J. Pravallika(20A81A1410), E. Suraj(20A81A1406) for the award of the degree of Bachelor of Technology in the Department of Electronics and Communication Technology during the academic year 2021-2024.

## Name of Project Guide Head of the Department

Mr. M. Subbarao Dr. E. Kusuma Kumari MTech., Ph.D.

Assistant. Professor Professor & HOD.

**DECLARATION**

We hereby declare that the project report entitled “**ENCRYPTION AND DECRYPTION**

**OF IMAGES**" submitted by us to Sri Vasavi Engineering College (Autonomous),

Tadepalligudem, affiliated to JNTUK Kakinada in partial fulfilment of the requirement for the award of the degree of B.Tech in Electronics and Communication Tchnology is a record of

Bonafide project work carried out by us under the guidance of **Mr. M. SUBBA RAO, ASST.PROFESSOR**. We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree in this institute or any other institute or University.

**Project Associates**

D. Saritha(20A81A1405)

K. Mohini Durga Bhavani

(20A81A1414)

M. Venkata Sai(20A81A1430)

J. Pravallika(20A81A1410)

E. Suraj(20a81A1406)

# ACKNOWLEDGEMENT

We would like to thank our project guide Mr. M. Subbarao, Department of ECE for the guidance and help throughout this project by providing us the required information.

We would like to thank Smt. Dr. E. KUSUMA KUMARI, Head of the Department of ECE, Dr. G.V.N.S.R. Ratnakar Rao, Principal of our college and Management of SRI VASAVI ENGINEERING COLLEGE for permitting us to do this project.

An endeavor can only be successful by constant effort and encouragement. We wish to take this opportunity to express our deep sense of gratitude to all the people who have extended their cooperation in various ways during this project. We take pleasure to acknowledge the help of all the respected elders.

**Project Associates**

D. Saritha(20A81A1405)

K. Mohini Durga Bhavani

(20A81A1414)

M. Venkata Sai(20A81A1430)

J. Pravallika(20A81A1410)

E. Suraj(20a81A1406)

|  |  |  |
| --- | --- | --- |
| SNO | TITLE | **PAGE NO** |
|  | **ABSTRACT** | 4 |
| 1 | INTRODUCTION 1.1 Introduction   * 1. Motivation   2. Scope | 5-7 |
| 2 | LITERATURE SURVEY 2.1 Literature Survey | 8-9 |
| 3 | INTRODUCTION TO PYTHON 3.1 What is Python?  3.2 Why Python is used?  3.3 About GUI Programming and Open CV  3.4 Technology Requirements | 10-15 |
| 4 | METHODOLOGY 4.1 Block Diagram  4.2 System Architecture  4.3 Implementation | 16-29 |
| 5 | TESTING 5.1 Introduction to Testing  5.2 Testing Strategies  5.3 Validation | 30-32 |
| 6 | RESULTS | 33-36 |

# TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| 7 | **CONCLUSION** | **37-38** |
| **8** | **REFERENCES** | **39-40** |

## ABSTRACT

In today’s environment, security becomes an important issue in communication. For secure transmission of data in open network, encryption is very important methodology. Through encryption we can prevent our data from unauthorized access during transmission. In recent years many image encryption methods have been proposed and used to protect confidential data and images. In this paper, we survey on existing work which used different techniques for image encryption. As security information has become the major concern for the fast growth of the digital exchange of data storage and transmission. As there is rapid growth of using images in many fields, so it is important to protect the private image data from the intruders. Image protection has become an imperative issue. To protect an individual privacy has become a crucial task. Different methods have been explored and developed to preserve data and personal information. To protect the important information from unauthorized users, image encryption is used. Encryption is the one of the most used techniques to hidden the data/image from unauthorized users. An Image Encryption Decryption is an image processing application created in python with Tkinter and OpenCV library.

**CHAPTER-1**

**INTRODUCTION**

### 1.1 INTRODUCTION

In the last few years, the security and integrity of the data is the most important concern. Now a day’s almost all the data is transferred over the computer networks and it has increased the attacks over the network. Before transmitted data it has to be encrypted and store so that it cannot be attacked by various attackers. Encryption is a process of hiding the data, where it converts the original text into cipher text. Encryption uses different algorithm to encrypt the data into different form.

Theimage preprocessing is the combination of many algorithms or methods to obtain desired output. Preprocessing is a method of transforming a source image to new image. This preprocessed image is almost similar to the source image, but differs in certain quality aspects like improved contrast, reduced noise, de-blur etc. Image processing through GUI makes the things simple and faster rather than entering commands. Preprocessed image gives better accuracy in pattern recognition and in classification.

**1.2. MOTIVATION**

There are many existing systems we have seen that encrypts and decrypts the images based on various complex algorithms. And almost all algorithms use key as a user input for encryption and decryption. So, there might be a chance of losing the key. In order to make the user’s intension of encrypting or decryption easier, we have decided to implement our project to provide an easy and secure Image Encryption Decryption Application.

**1.3. SCOPE**

The scope of our project is to Encrypt and Decrypt Images using Python tkinter GUI and Open-CV library.

# CHAPTER-2

**LITERATURE SURVEY**

**2.1 LITERATURE SURVEY**

**New Mirror-Like Image Encryption Algorithm and Its VLSI Architecture.**

Jiun-In Guo and Jui-Cheng Yen [3] have presented an algorithm which was mirror like. In this algorithm there were 7 steps. In the first, 1-D chaotic system is determined and its init ial point x (0) and sets k = 0. Then, the chaotic sequence is generated from the chaotic system. After that binary sequence is generated from chaotic system. And in last 4 stages image pixels are rearranged using swap function according to the binary sequence.

1. **Lossless Image Compression and Encryption Using SCAN.**

S.S. Maniccam and N.G. Bourbakis [4] have presented a new algorithm which does two works: lossless compression and encryption of binary and gray-scale pictures. The compression and encryption schemes are based on SCAN patterns generated by the SCAN methodology. The SCAN is formal language-based 2D spatial-accessing methodologies generate a wide range of scanning paths or space filling curves.

1. **New Encryption Algorithm for Image Cryptosystems.**

Chin-Chen Chang, Min-Shian Hwang, and Tung-Shou Chen [6] used vector quantization for designing better cryptosystem for images. The scheme was based on vector quantization (VQ), cryptography, and various others number theorem. In vector quantization (VQ) firstly the images are decomposed into vectors and then sequentially encoded vector by vector.

1. **Technique for Image Encryption using Digital Signatures.**

Aloka Sinha and Kehar Singh [4] have proposed a new technique in which the digital signature of the original image is added to the encoded version of the original image. A best suitable error code is followed to do encoding of the image, ex: Bose-Chaudhuri Hochquenghem (BCH) code. At the receiver end, after decryption of that image, the digital signature verifies the authenticity of the image.

1. **Technique for Image Encryption using multi-level and image dividing technique.**  Chang-Mok Shin, Dong-Hoan Seo, Kyu-Bo Chol, Ha Wmn Lee, and SmJmng Kim [7] proposed an algorithm which was multilevel form of image encryption using binary phase exclusive OR operation and image dividing technique. The same grey level multi-level image is divided into binary images. Then binary pictures are regenerate to binary phase encoding and then these images are encrypt with binary random phase images by binary phase XOR operation.

1. **Technique for Image Encryption using 1D chaotic map.**

Fethi Belkhouche and Uvais Qidwai [8] used the method that can be used for binary images encryption with the possibility of using several keys ex: initial state, the external parameters and iterations’ number.

1. **Technique for Image Encryption using chaos technique.**

Guo sheng Gu and Guoliang Han [10] made a new highly optimized image algorithm using permutation and substitution methods. It was done in order to enhance the pseudorandom characteristics of chaotic sequences; an optimized treatment and a cross-sampling disposal is used.

# CHAPTER-3

**INTROUCTION TO PYHON**

**3.1 WHAT IS PYTHON?**

Python is developed by Guido van Rossum. Guido van Rossum started implementing Python in 1989. Python is a very simple programming language so even if you are new to programming, you can learn python without facing any issues. Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting language to connect existing components together. Python is a simple, easy-to-learn syntax that emphasizes readability and therefore reduces the cost of programming maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms and can be freely distributed. Often programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-debug-test cycle is incredibly fast. Debugging Python programs is easy because a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn’t catch the exception, the interpreter prints a stack trace. A source-level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, and stepping through the code a line at a time.

**3.2 WHY PYTHON IS USED?**

Python language is one of the most accessible programming languages available because it has simplified syntax and is not complicated, which gives more emphasis on natural language. Due to its ease of learning and usage, python codes can be easily written and executed much faster than other programming languages. Python has a lot of modules, they are essentially python script files that can contain variables, functions, and classes. Python modules help us in organizing our code and then referencing them in other classes or python scripts. Python has a robust and large standard library that makes it stand out from other programming languages. Its standard library contains a wide range of modules, operations and web service tools that you can select and use for your applications without writing code

Data visualization

PlotLib

WEB Frame Work

Django

**3.3 ABOUT OPEN-CV and GUI PROGRAMMING**

**OpenCV-python**

OpenCV-Python is a library of Python bindings designed to solve computer vision problems. Python is a general-purpose programming language started by Guido van Rossum that became very popular very quickly, mainly because of its simplicity and code readability. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in commercial products. OpenCV python library allows you to perform image processing and computer vision tasks. It provides a wide range of features, including object detection, face recognition, and tracking. Python is well-suited for implementing new features. cv2 (old interface in old OpenCV versions was named as cv ) is the name that OpenCV developers chose when they created the binding generators. This is kept as the import name to be consistent with different kind of tutorials around the internet. OpenCV provides a module called ml that has many machine learning algorithms bundled into it. Some of the algorithms include Bayes Classifier, K-Nearest Neighbors, Support Vector Machines, Decision Trees, Neural Networks, and so on. We can have a lot of modules in python with simple syntax and output can be obtained in less lines of code.

**GUI Inbuilt Functions:**

* + 1. **bd**: to set the border width in pixels.
    2. **bg**: to set the normal background color.
    3. **cursor**: to set the cursor used.
    4. **command**: to call a function.
    5. **highlightcolor**: to set the color shown in the focus highlight.
    6. **width**: to set the width of the button.
    7. **height**: to set the height of the button.

**GUI PROGRAMMING (Tkinter):**

Python provides various options for developing graphical user interfaces (GUIs). Most important are listed below.

* Tkinter − Tkinter is the Python interface to the Tk GUI toolkit shipped with Python.
* wxPython − This is an open-source Python interface for wxWindows
* JPython − JPython is a Python port for Java which gives Python scripts seamless access to Java class libraries on the local machine.

**Tkinter Programming:**

Python has a lot of [GUI frameworks,](http://wiki.python.org/moin/GuiProgramming) but [Tkinter](https://wiki.python.org/moin/TkInter) is the only framework that’s built into the Python standard library. Tkinter has several strengths. It’s **cross-platform**, so the same code works on Windows, macOS, and Linux. Visual elements are rendered using native operating system elements, so applications built with Tkinter look like they belong on the platform where they’re run.

Although Tkinter is considered the de facto Python GUI framework, it’s not without criticism. One notable criticism is that GUIs built with Tkinter look outdated. If you want a shiny, modern interface, then Tkinter may not be what you’re looking for.

However, Tkinter is lightweight and relatively painless to use compared to other frameworks. This makes it a compelling choice for building GUI applications in Python, especially for applications where a modern sheen is unnecessary, and the top priority is to quickly build something that’s functional and cross-platform.

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful objectoriented interface to the Tk GUI toolkit.

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps −

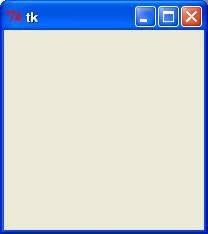
* Import the *Tkinter* module.
* Create the GUI application main window.
* Add one or more of the above-mentioned widgets to the GUI application.
* Enter the main event loop to take action against each event triggered by the user.

**Example:**

import Tkinter top = Tkinter.Tk()

# Code to add widgets will go here... top.mainloop()

This would create a following window −



**3.4 TECHNOLOGY REQUIREMENTS**

* Python 3
* Visual Studio Code
* os module
* cv2 module
* tkinter module
* filedialog from tkinter
* messagebox
* from PIL import Image, ImageTk
* numpy
* random

**CHAPTER-4**

**METHODOLOGY**

**4.1 Block Diagram**

Choose an image

Click on encrypt button

It encrypts the image successfully into grey scale

Click on Decrypt button

It decrypts the image successfully

Into grey scale image

Click on Reset button

It resets into original form

**4.2 SYSTEM ARCHITECTURE**

A system Architecture is the conceptual model that defines the structure, behaviour, and more views of the system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviours of the system.

Our project design can be explained and understood clearly by studying the OpenCV and GUI inbuilt functions

1. **Color to GrayScale**

- We convert the BGR images to grayscale images so that we can observe the intensity of the pixels in the image rather than the color configurations in the image.

**Code Implementation:**

import cv2 as cv

img = cv.imread('PATH OF THE IMAGE IN YOUR LOCAL SYSTEM')

cv.imshow("Color",img)

gray =

cv.cvtColor(img,cv.COLOR\_BGR2GRAY)

cv.imshow("Gray",gray)

cv.waitKey(0)

Fig:RGB image



**Results:**



Fig:GrayScale image.

1. **Blurring an image using GuassianBlur**.

We blur an image in order to reduce the noise in the image caused by bad lighting and some issues with the sensors in cameras while capturing the image. In the code, in order to increase the range of blur, we can increase the kernel size from (3,3) to (7,7).

**Code Implementation:**

|  |
| --- |
| import cv2 as cv img = cv.imread('Path to the image in your local system') cv.imshow("Color",img)+  blur = cv.GaussianBlur(img,(7,7),cv.BORDER\_DEFAULT) cv.imshow("Blurred",blur) cv.waitKey(0) |



Fig:Before Blurring

**Results:**



Fig:Blurred image

1. **Edge Cascade**

to detect edges in an image for better model learning.We using the Canny Edge detector which is very popular.

**Code Implementation:**

import cv2 as cv

img =cv.imread('Path to the image in your local system')

cv.imshow("Color",img) canny = cv.Canny(img,125,175) cv.imshow("Edge Cascaded image",canny) cv.waitKey(0)



Fig:RGB image

**Results:**

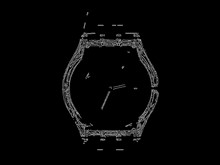


Fig:Edge detection

1. **Dilation of the cascaded image**

This thickens the edges from the cascaded image to enhance edge detectability and better detect features.

**Code Implementation:**

import cv2 as cv

img = cv.imread('Path to the image in your local system')

cv.imshow("Color",img) canny = cv.Canny(img,125,175) #dilating the images: dilated = cv.dilate(canny,(7,7),iterations=3) cv.imshow("dilated images",dilated) cv.waitKey(0)

**Results:**

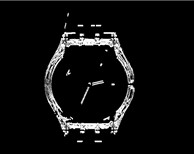


Fig:Dilated image

1. **Resize and cropping the image**

for the same dimensions for images of different dimensions.

**Code Implementation:**

|  |
| --- |
| import cv2 as cv  img = cv.imread('Path to the image in your local system') cv.imshow("Color",img) #resizing the images:  resize = cv.resize(img,(500,500),interpolation=cv.INTER\_CUBIC) crop = img[50:200,200:400] cv.imshow("cropped image",crop) cv.imshow("Resized images",resize)cv.waitKey(0)    **Results:** |

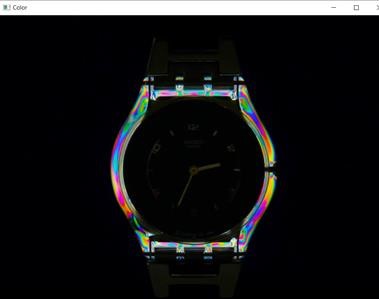


Fig:Resized image

1. **Determining contours in an image.**

Contours are nothing but a curve or line joining all the continuous points in the image(along the borders)which has the same color or intensity. It is used in object detection mainly for shape analysis.

**Code implementation:**

|  |
| --- |
| import cv2 as cv  img = cv.imread('Path to the image in your local system') cv.imshow("Color",img)  #reduces the contours when blurring the image  blur = cv.GaussianBlur(img,(7,7),cv.BORDER\_DEFAULT) canny = cv.Canny(blur,125,175)  #returns all the contours in the forma of list(RETR\_LIST).  #chain\_approx\_simple compresses the contorurs and returns only the two end points.  contours,hierarchies = cv.findContours(canny,cv.RETR\_LIST,cv.CHAIN\_APPROX\_SIMPLE) print("NO of contours : ",len(contours))  #threshold = binarizing the image.It is another method to find contours ret,thresh = cv.threshold(img,125,125,cv.THRESH\_BINARY) cv.imshow("Thresh",thresh)  cv.waitKey(0) **Results:** |

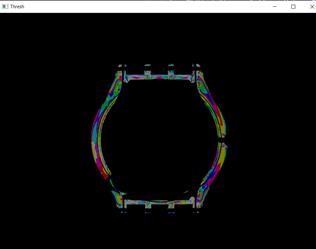


Fig:Contoured image. We can see just the borders.

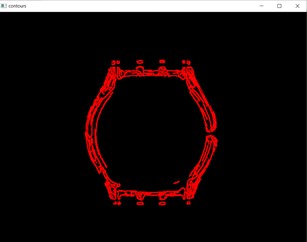


Fig:Drawing and visualizing the contours.

1. **Splitting an image into its respective RED, GREEN, and BLUE parts**.

Here it is depicted as a GreyScale image and therefore we can observe the concentration of pixels for respective colors.

**Code Implementation:**

|  |
| --- |
| import cv2 as cv import numpy as np  img = cv.imread('Path to the image in your local system') cv.imshow("Color",img)  b,g,r = cv.split(img) cv.imshow("Blue",b) cv.imshow("Green",g)  cv.imshow("Red",r)    **Results:** |

Fig:

Blue split

Fig:

Green split

Fig:

Red split



**4.3 Implementation Steps** **:**

* Install tkinter, PIL, numpy, cv2, os, random
* After that upload the code file, and run image\_encryption\_decryption.py on local system.
* Then the script will start running and user can explore it by encrypting and decrypting any image and saving it.

**How to use it:**

* User just need to download the file, and run the image\_encryption\_decryption.py, on local system.
* After running a GUI window appears, where user needs to choose an image file using CHOOSE button on the top right corner.
* After selecting the image, two images will appear on screen one on left side, which is original and one on write in which Encrypted Decrypted format will be shown.
* Now user can start encryption and decryption using Encrypt and Decrypt button.
* After editing user can also save the edited image to any location in local system using SAVE button.
* Also there is a RESET button, clicking on which resets the edited image to original format.
* Also there is exit button, clicking on which we get a exit dialog box asking the permission to exit.

**6.2 Coding**

# imported necessary library import tkinter from tkinter import \* import tkinter as tk

import tkinter.messagebox as mbox

from tkinter import ttk from tkinter import filedialog from PIL import ImageTk, Image import cv2 import os import numpy as np from cv2 import \* import random

import time

#created main window window = Tk() window.geometry("1000x700") window.title("Image Encryption and Decryption") window.configure(bg='light green') time.sleep(1)

# defined variable global count, emig # global bright, con

# global frp, tname # list of paths

frp = [] tname = [] con = 1 bright = 0 panelB = None

panelA = None

# function defined to get the path of the image selected def getpath(path): a = path.split(r'/') # print(a) fname = a[-1] l = len(fname) location = path[:-l]

return location

# function defined to get the folder name from which image is selected def getfoldername(path): a = path.split(r'/') # print(a) name = a[-1]

return name

# function defined to get the file name of image is selected def getfilename(path): a = path.split(r'/') fname = a[-1] a = fname.split('.') a = a[0]

return a

# function defined to open the image file def openfilename(): filename = filedialog.askopenfilename(title='"pen') return filename

# function defined to open the selected image def open\_img(): global x, panelA, panelB global count, eimg, location, filename

count = 0 x = openfilename() img = Image.open(x) eimg = img

img = ImageTk.PhotoImage(img)

temp = x location = getpath(temp) filename = getfilename(temp)

# print(x) if panelA is None or panelB is None:

panelA = Label(image=img) panelA.image = img

panelA.pack(side="left", padx=10, pady=10) panelB = Label(image=img) panelB.image = img panelB.pack(side="right", padx=10, pady=10) else:

panelA.configure(image=img) panelB.configure(image=img) panelA.image = img

panelB.image = img

# function defined for make the sketch of image selected def en\_fun(): global x, encrypted\_image, key

# print(x)

input\_image = cv2.imread(x, 0)# 'C:/Users/aakas/Documents/flower.jpg'

(x1, y) = input\_image.shape

input\_image = input\_image.astype(float) / 255.0

# print(image\_input)

mu, sigma = 0, 0.1 # mean and standard deviation

key = np.random.normal(mu, sigma, (x1, y)) + np.finfo(float).eps

# print(key)

encrypted\_image = input\_image / key

cv2.imwrite('encrypted\_image.jpg', encrypted\_image \* 255)

imge = Image.open('encrypted\_image.jpg') imge = ImageTk.PhotoImage(imge) panelB.configure(image=imge) panelB.image = imge

mbox.showinfo("Encrypt Status", "Image Encryted successfully.")

# function defined to make the image sharp def de\_fun(): global encrypted\_image, key output\_image = encrypted\_image \* key output\_image \*= 255.0

cv2.imwrite('output\_image.jpg', output\_image)

imgd = Image.open('output\_image.jpg') imgd = ImageTk.PhotoImage(imgd) panelB.configure(image=imgd) panelB.image = imgd

mbox.showinfo("Decrypt Status", "Image decrypted successfully.")

# function defined to reset the edited image to original one def reset(): # print(x)

image = cv2.imread(x)[:, :, ::-1] global count, eimg count = 6 global o6 o6 = image

image = Image.fromarray(o6) eimg = image image = ImageTk.PhotoImage(image) panelB.configure(image=image) panelB.image = image

mbox.showinfo("Success", "Image reset to original format!")

# function defined to same the edited image def save\_img(): global location, filename, eimg print(filename)

# eimg.save(location + filename + r"\_edit.png") filename = filedialog.asksaveasfile(mode='w', defaultextension=".jpg") if not filename:

return

eimg.save(filename)

mbox.showinfo("Success", "Encrypted Image Saved Successfully!")

# top label start1 = tk.Label(text = "Image Encryption\nand\nDecryption", font=("Arial", 25),bg='light

green', fg="black") # same way bg

start1.place(x = 350, y = 10)

# original image label start1 = tk.Label(text = "Original\nImage", font=("Arial", 25),bg='light green', fg="black") #

same way bg

start1.place(x = 100, y = 270)

# edited image label start1 = tk.Label(text = "Encrypted\nDecrypted\nImage", font=("Arial",25),bg='light green',

fg="black") # same way bg

start1.place(x = 700, y = 230)

# choose button created chooseb = Button(window, text="Choose",command=open\_img,font=("Arial", 15), bg =

"black", fg = "pink", borderwidth=3, relief="raised") chooseb.place(x =30 , y =20 )

# save button created saveb = Button(window, text="Save",command=save\_img,font=("Arial", 15), bg = "black", fg

= "pink", borderwidth=3, relief="raised") saveb.place(x =170 , y =20 )

# Encrypt button created enb = Button(window, text="Encrypt",command=en\_fun,font=("Arial", 15), bg = "black", fg

= "pink", borderwidth=3, relief="raised") enb.place(x =150 , y =620 )

# decrypt button created deb = Button(window, text="Decrypt",command=de\_fun,font=("Arial", 15), bg = "pink", fg =

"black", borderwidth=3, relief="raised") deb.place(x =450 , y =620 )

# reset button created resetb = Button(window, text="Reset",command=reset,font=("Arial", 15), bg = "black", fg =

"pink", borderwidth=3, relief="raised") resetb.place(x =800 , y =620 )

# function created for exiting def exit\_win(): if mbox.askokcancel("Exit", "Do you want to exit?"):

window.destroy()

# exit button created exitb = Button(window, text="EXIT",command=exit\_win,font=("Arial", 15), bg = "red", fg =

"white", borderwidth=3, relief="raised") exitb.place(x =880 , y =20 )

window.protocol("WM\_DELETE\_WINDOW", exit\_win) window.mainloop()

**CHAPTER-5**

**TESTING**

## 5.1 INTRODUCTION TO TESTING

Testing is a process, which reveals errors in the program. It is the major quality measure employed during software development. During testing, the program is executed with a set of test cases and the output of the program for the test cases is evaluated to determine if the program is performing as it is expected to perform.

## 5.2 TESTING STRATEGIES

In order to make sure that the system does not have errors, the different levels of testing strategies that are applied at differing phases of software development are:

### Unit Testing

Unit Testing is done on individual modules as they are completed and become executable. It is confined only to the designer's requirements. Each module can be tested using the following Strategies:

**Black Box Testing:**

In this strategy some test cases are generated as input conditions that fully execute all functional requirements for the program. This testing has been uses to find errors in the following categories:

* Incorrect or Missing Functions
* Interface Errors
* Errors in data structure or external database access
* Performance errors
* Initialization and termination errors

In this testing only the output is checked for correctness. The logical flow of the data is not checked.

### Integrating Testing Integration

Testing ensures that software and subsystems work together a whole. It tests the interface of all the modules to make sure that the modules behave properly when integrated together. In this case the communication between the device and Google Translator Service.

### System Testing

Involves in-house testing in an emulator of the entire system before delivery to the user. Its aim is to satisfy the user the system meets all requirements of the client's specifications.

### Acceptance Testing

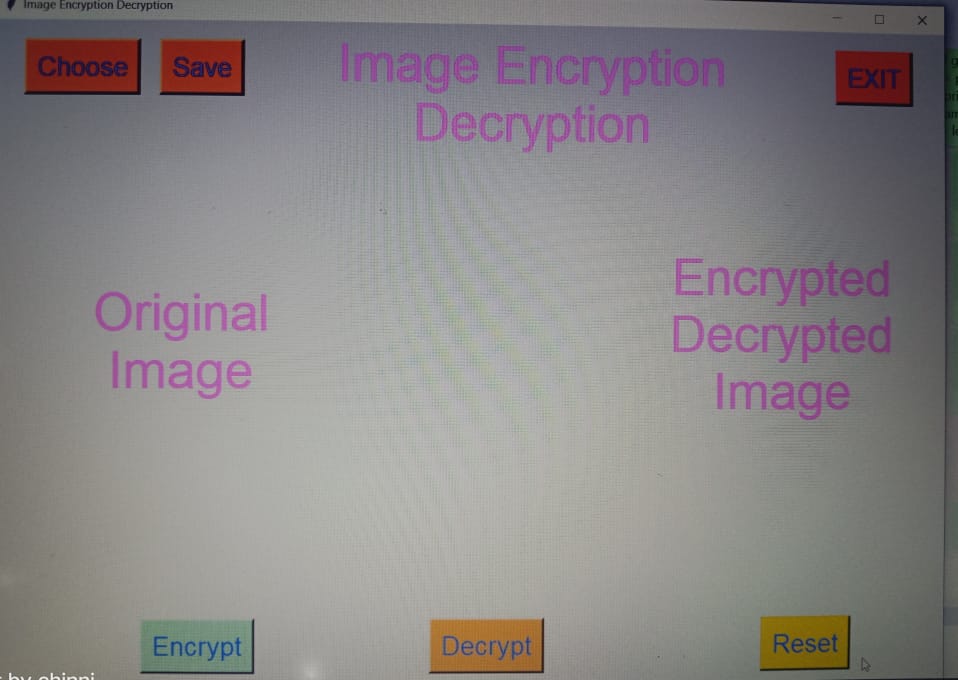
It is a pre-delivery testing in which entire system is tested in a real android device on real world data and usage to find errors.

## 5.3 VALIDATION

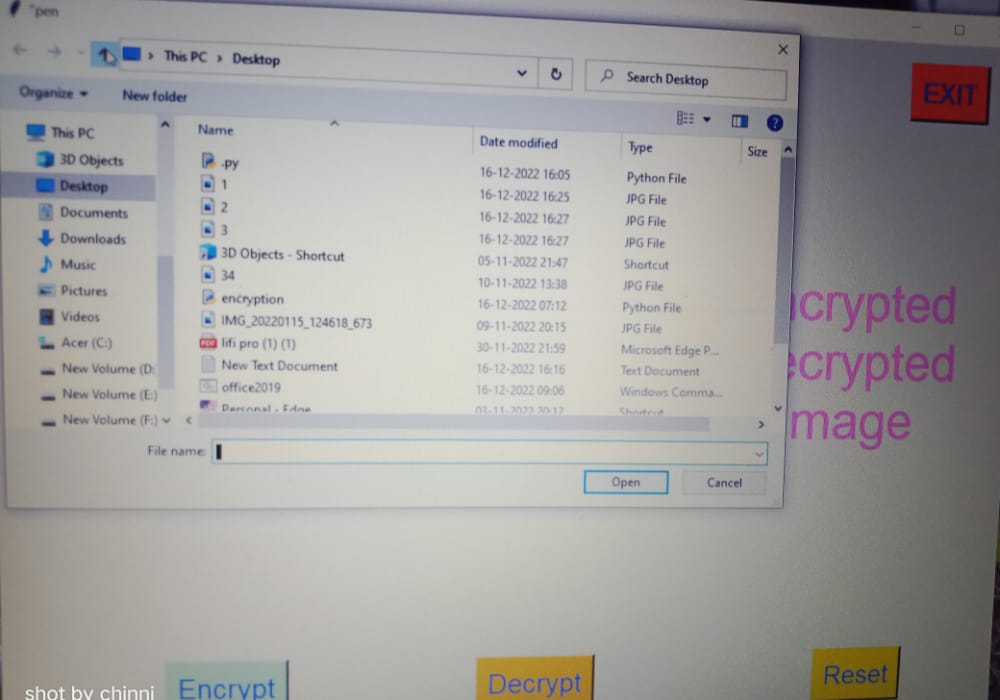
The system has been tested and implemented successfully and thus ensured that all the requirements as listed in the software requirements specification are completely fulfilled. In case of erroneous input corresponding error messages are displayed.

**CHAPTER-6**

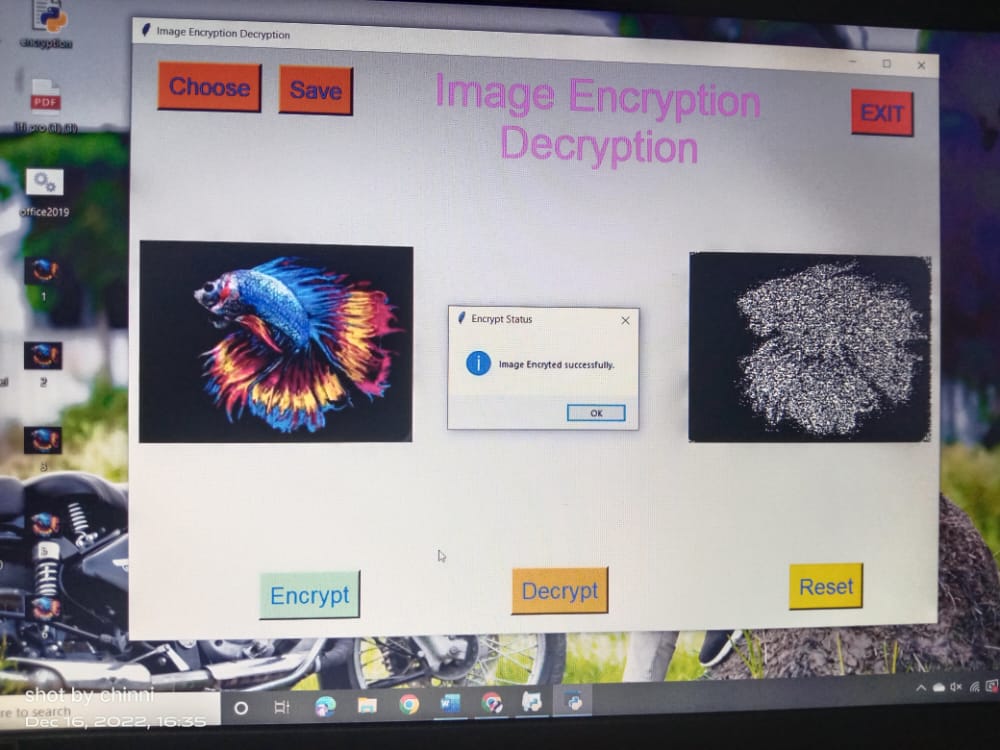
# RESULTS



**Fig 1. Display of Main window**



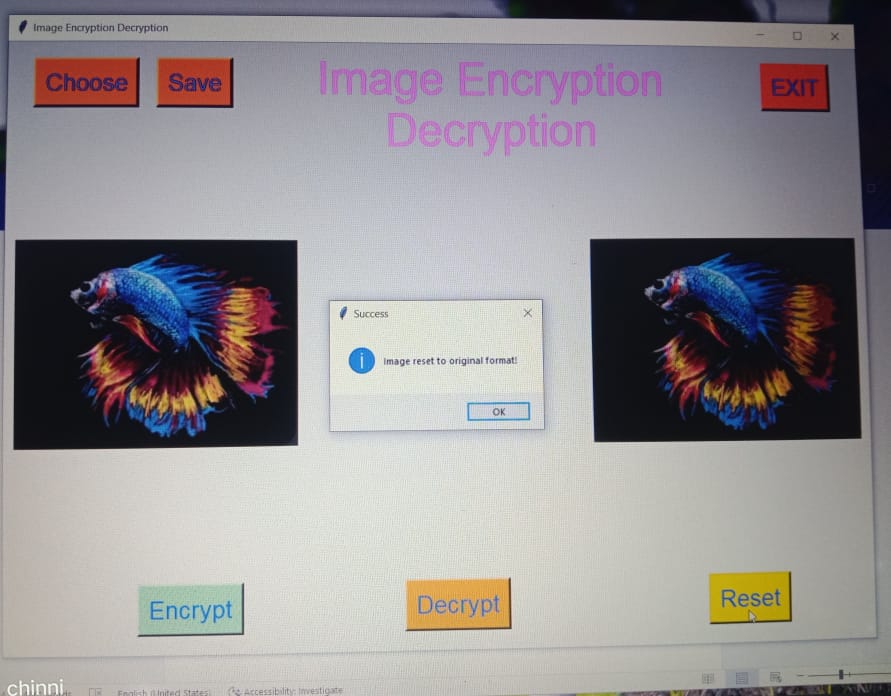
**Fig 2. Choose an Image**



## Fig 3. Image Encryption



**Fig 4. Image Decryption**



**Fig 5. Image reset to original format**

# CHAPTER-7

# CONCLUSION

## 9.1 Result and Conclusion

The image which has to be encrypted is chosen from the folder and the encrypt button is clicked. The original input image taken in the form of .JPG file as shown in fig 2. Once the image is encrypted successfully then the message is displayed as in fig 3. And when the image is viewed it shows that the image is encrypted. For decryption of the image which has been encrypted then the encrypted image has to be select and then decrypted button is clicked, then the decryption will be done and a message of successful decryption is shown as in fig 4. When the image is reset then it shows the original image as shown in fig 5. And then click on exit button to close the window.

# CHAPTER 8

# REFERENCES

## 10.1 REFERENCES

1. Abdel-Karim, “Performance Analysis of Data Encryption Algorithms”. [**Google Scholar**](http://scholar.google.com/scholar?hl=en&q=Abdel-Karim%2C+%E2%80%9C+Performance+Analysis+of+Data+Encryption+Algorithms%E2%80%9D.)
2. Y. Ou, C. Sur, K. H. Rhee “Region based selective Encryption for Medical Imaging”, 1st Annual International Workshop-2007, [**Google Scholar**,](http://scholar.google.com/scholar_lookup?hl=en&publication_year=2007&conference=1st+Annual+International+Workshop-&author=Y.+Ouauthor=C.+Surauthor=K.+H.+Rhee&title=Region+based+selective+Encryption+for+Medical+Imaging) [**Crossref.**](https://aip.scitation.org/servlet/linkout?suffix=c2/c2_1&dbid=16&doi=10.1063%2F1.5032036&key=10.1007%2F978-3-540-73814-5_6)
3. S. H. Kamali, R. Shakerian, M. Hedayati, “A new modified version of Advanced Encryption Standard based algorithm for image encryption”, International Conference on Electronics and information Engineering, ICEIE-2010. [**Google Scholar**](http://scholar.google.com/scholar?hl=en&q=S.+H.+Kamali%2C+R.+Shakerian%2C+M.+Hedayati%2C+%E2%80%9C+A+new+modified+version+of+Advanced+Encryption+Standard+based+algorithm+for+image+encryption%E2%80%9D%2C+International+Conference+on+Electronics+and+information+Engineering%2C+ICEIE-2010.)
4. Nur Nabila Mohamed, Habibah Hashim, Yusnani Mohd Yussoff, “Compression and Encryption Technique on Securing TFTP Packet”, 2014 IEEE Symposium on Computer

Applications & Industrial Electronics (ISCAIE), April 7-8, 2014, Penang, Malaysia. [**Google Scholar**](http://scholar.google.com/scholar?hl=en&q=Nur+Nabila+Mohamed%2C+Habibah+Hashim%2C+Yusnani+Mohd+Yussoff%2C+%E2%80%9C+Compression+and+Encryption+Technique+on+Securing+TFTP+Packet%E2%80%9D%2C+2014+IEEE+Symposium+on+Computer+Applications+%26+Industrial+Electronics+%28ISCAIE%29%2C+April+7-8%2C+2014%2C+Penang%2C+Malaysia.)

1. H. Trang and N. Loi, “An efficient FPGA implementation of the Advanced Encryption Standard,” in Computing and Communication Technalogies, Research, Innavation, and

Visian far the Future (RIVF), 2012 IEEE RIVF Internatianal Conference on, March 2012. [**Google Scholar**](http://scholar.google.com/scholar_lookup?hl=en&publication_year=2012&conference=Computing+and+Communication+Technalogies%2C+Research%2C+Innavation%2C+and+Visian+far+the+Future+%28RIVF%29%2C+2012+IEEE+RIVF+Internatianal+Conference+on&author=H.+Trangauthor=N.+Loi&title=An+efficient+FPGA+implementation+of+the+Advanced+Encryption+Standard)

1. M.P. Priyanka, E. Lakshmi, Dr.A.R. Reddy, “FPGA Implementation Of Image Encryption And Decryption Using AES 128-Bit Core”, Communication and Electronics Systems (ICCES), International Conference on 21-22 Oct. 2016, [**Google Scholar**,](http://scholar.google.com/scholar_lookup?hl=en&publication_year=2016&conference=Communication+and+Electronics+Systems+%28ICCES%29%2C+International+Conference+on+21-22+Oct&author=M.P.+Priyankaauthor=E.+Lakshmiauthor=A.R.+Reddy&title=FPGA+Implementation+Of+Image+Encryption+And+Decryption+Using+AES+128-Bit+Core) [**Crossref.**](https://aip.scitation.org/servlet/linkout?suffix=c6/c6_1&dbid=16&doi=10.1063%2F1.5032036&key=10.1109%2FCESYS.2016.7889929)
2. El Maraghy M, Hesham S and Abd El Ghany M.A., “Real-time Efficient FPGA Implementation of AES Algorithm”, IEEE International SOC Conference (SOCC), Sept 2013. [**Google Schola**](http://scholar.google.com/scholar_lookup?hl=en&publication_year=2013&conference=IEEE+International+SOC+Conference+%28SOCC%29&author=M+El+Maraghyauthor=S+Heshamauthor=M.A.+Abd+El+Ghany&title=Real-time+Efficient+FPGA+Implementation+of+AES+Algorithm)[**rCrossref**](https://aip.scitation.org/servlet/linkout?suffix=c7/c7_1&dbid=16&doi=10.1063%2F1.5032036&key=10.1109%2FSOCC.2013.6749688)
3. Monica Lib eratori, Fernando Otero, J. C. Bonadero, Jorge Castifieira “AES-128 cipher. high speed, low cost FPGA implementation”, IEEE-2007. [**Google Scholar**](http://scholar.google.com/scholar_lookup?hl=en&publication_year=2007&author=Monica+Lib+eratoriauthor=Fernando+Oteroauthor=J.+C.+Bonaderoauthor=Jorge+Castifieira&title=AES-128+cipher.+high+speed%2C+low+cost+FPGA+implementation)
4. Chi-Wu Huang, Chi-Jeng Chang, Mao-Yuan Lin, Hung-Ym Tai, “Compact FPGA Implementationof 32-bits AES Algorithm Using Block RAM”, IEEE-2007. [**Google Scholar**](http://scholar.google.com/scholar_lookup?hl=en&publication_year=2007&author=Chi-Wu+Huangauthor=Chi-Jeng+Changauthor=Mao-Yuan+Linauthor=Hung-Ym+Tai&title=Compact+FPGA+Implementationof+32-bits+AES+Algorithm+Using+Block+RAM)
5. N. N. Mohamed, H. Hashim, Y. M. Yussoff, and A. M. Isa, “Securing TFTP packet: A preliminary study,” IEEE 4th Control Syst. Grad. Res. Colloq. Aug. 2013. [**Google Scholar**,](http://scholar.google.com/scholar_lookup?hl=en&publication_year=2013&conference=IEEE+4th+Control+Syst.+Grad.+Res.+Colloq.+Aug&author=N.+N.+Mohamedauthor=H.+Hashimauthor=Y.+M.+Yussoffauthor=A.+M.+Isa&title=Securing+TFTP+packet%3A+A+preliminary+study) [**Crossref.**](https://aip.scitation.org/servlet/linkout?suffix=c10/c10_1&dbid=16&doi=10.1063%2F1.5032036&key=10.1109%2FICSGRC.2013.6653295)
6. [http://tools.ietf.org/pdf/rfc783.pdf.](http://tools.ietf.org/pdf/rfc783.pdf) [**Google Scholar**](http://scholar.google.com/scholar?hl=en&q=http%3A%2F%2Ftools.ietf.org%2Fpdf%2Frfc783.pdf.) 12. [http://tools.ietf.org/pdf/rfc2347.pdf.](http://tools.ietf.org/pdf/rfc2347.pdf) [**Google Scholar**](http://scholar.google.com/scholar?hl=en&q=http%3A%2F%2Ftools.ietf.org%2Fpdf%2Frfc2347.pdf.)