A

Mini

Project

On

On the Implementation of a Secured Watermarking Mechanism Based on Cryptography and Bit Pairs Matching

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

Syamala Saketh Raja Reddy (217R1A0555)

Karnam Rahul(217R1A0531)

Kallem Nithin Reddy (217R1A0529)

Under the Guidance of

Dr.K.Maheswari

(Associate Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

2021-25

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled "**On the Implementation of a Secured Watermarking Mechanism Based on Cryptography and Bit Pairs Matching**" being submitted by **Syamala Saketh Raja Reddy (217R1A0555) ,Karnam Rahul (217R1A0531) ,Kallem Nithin Reddy (217R1A0529)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by him/her under our guidance and supervision during the year 2024-25.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Dr.K.Maheswari
Associate Professor
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. Nuthanakanti Bhaskar
HOD of CSE

EXTERNAL EXAMINER

Submitted for viva voice Examination held on _____

ACKNOWLEGDEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project. We take this opportunity to express my profound gratitude and deep regard to my guide

Dr. K Maheswari, Associate Professor for her exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by her shall carry us a long way in the journey of life on which we are about to embark. We also take this opportunity to express a deep sense of gratitude to Project Review Committee (PRC) Coordinators: **Dr. K Maheswari, Dr. J.Narasimha Rao, Mrs.K.Shilpa, Mr.K.Ranjith Reddy** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. Nuthanakanti Bhaskar,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We would like to express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

Syamala Saketh Raja Reddy (217R1A0555)
Karnam Rahul (217R1A0531)
Kallem Nithin Reddy (217R1A0529)

ABSTRACT

Watermarking is one of the most vital digital information hiding technique, which can be used with cryptography mechanism for providing more security to digital data. In image watermarking mechanism mostly LSB substitution is used on the cover image for hiding the secret watermark. In this paper, a novel technique based on the matching of bit pairs and symmetric key cryptography is proposed. Pixel bits of original image and encrypted watermark image are arranged in pairs. The pixel bits are represented in pairs following the proposed algorithm, then the encrypted watermark pixel bit pairs are compared with all bit pairs of original image and accordingly the replacement of bit pairs takes place with the respective matched pair assigned number binary equivalent. If no match is found then go for replacing the 0th pair with watermark bits and replace the two LSB with the value of pair number 0. The proposed mechanism shows good quality of watermarked image along with good PSNR values with a good payload. By com- paring the results with some existing algorithms, the proposed scheme shows the valuable results.

LIST OF FIGURES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

1.INTRODUCTION

1.INTRODUCTION

1.1 PROJECT SCOPE

The scope of this paragraph is to provide an overview of digital watermarking techniques that are used for protecting multimedia information from unauthorized use. It discusses two primary categories of watermark embedding methods: spatial domain and frequency domain techniques. The focus is on the balance between robustness, computational cost, and imperceptibility across various types of multimedia, including images, audio, and video.

1.2 PROJECT PURPOSE

The purpose is to highlight the importance of robust watermarking techniques for copyright protection. It emphasizes the need to select watermarking methods that strike a balance between robustness and computational efficiency, particularly when facing potential attacks like noise addition, filtering, or compression. Additionally, the paragraph underscores key considerations such as imperceptibility, robustness, payload capacity, security, and trustworthiness, which are crucial in the implementation of invisible watermarking.

1.3 PROJECT FEATURES

The features of digital watermarking techniques for multimedia copyright protection include two main categories: spatial domain and frequency domain methods. Spatial domain techniques are faster and require less computational power but are less robust against attacks. Conversely, frequency domain methods offer enhanced robustness but come at a higher computational cost. Effective watermarking involves balancing robustness, which ensures the watermark can withstand attacks, with imperceptibility to maintain the original media's quality. Additionally, these techniques consider capacity, allowing sufficient information to be embedded, and security, often bolstered by cryptographic or steganographic methods. Trade-offs are inherent, as enhancing robustness might affect imperceptibility or computational efficiency. Approaches like LSB matching and complexity-based schemes are commonly employed to improve security and data hiding capacity

2.SYSTEM ANALYSIS

2.SYSTEM ANALYSIS

SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

2.1 PROBLEM DEFINITION

The growing need for copyright protection of multimedia content, such as images, audio, and video, has led to the development of various digital watermarking techniques. These techniques aim to safeguard multimedia information from unauthorized use. Digital watermarking typically involves embedding a watermark within the media, which can later be used to verify ownership or authenticity. The two primary categories of watermark embedding techniques are spatial domain and frequency domain methods. However, there is a trade-off between computational cost and robustness, which makes it challenging to maintain a balance between imperceptibility and robustness, while also achieving high payload capacity and security.

2.2 EXISTING SYSTEM

In Current digital watermarking techniques in use include Least Significant Bit (LSB) substitution, LSB matching, and Pixel Value Differencing (PVD). LSB substitution is a fast and widely used technique where the least significant bits of the cover image pixels are replaced with watermark data. PVD methods, on the other hand, focus on embedding data in the edge areas of an image, which tend to be more robust but can introduce visible distortions. LSB matching is a more secure approach than simple LSB substitution and involves finding a match between watermark and cover image bits to minimize distortion. Additionally, cryptographic methods are sometimes used alongside watermarking to further enhance security by encrypting the watermark before embedding it.

2.2.1 LIMITATIONS OF EXISTING SYSTEM

- Imperceptibility: Many methods can introduce visible distortions, particularly in spatial domain techniques, impacting the quality of the original image.
- Imperceptibility: Many methods can introduce visible distortions, particularly in spatial domain techniques, impacting the quality of the original image.
- Payload Capacity: The amount of data that can be embedded using current methods is often limited, especially in techniques like LSB substitution, which restricts the ability to embed larger or more complex watermarks
- Security: Simple LSB techniques, though fast, lack robustness against steganalysis attacks and do not provide sufficient security for sensitive information. Advanced techniques, such as those integrating cryptography, require more resources but can offer higher security and robustness.

To avoid all these limitations and make the working more accurately the system needs to be implemented efficiently.

2.3 PROPOSED SYSTEM

The proposed system is a digital watermarking method that uses symmetric key cryptography and bit pairs matching in the spatial domain to enhance the security and robustness of embedded watermarks. The process begins with encrypting the watermark using a symmetric key, ensuring that it is protected during transmission. The system then utilizes bit pairs matching, where the encrypted watermark is embedded into the original image by replacing the least significant bits (LSBs) of the cover image's pixels.

The embedding process involves matching bit pairs of the encrypted watermark with the cover image, selectively embedding them to maintain image quality. During extraction, the LSBs of the watermarked image are retrieved and used to reconstruct the encrypted watermark. Finally, the watermark is decrypted using the same symmetric key to restore the original image.

Figure 2.3.1: Original Grayscale Images

2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

The system is very simple in design and to implement. The system requires very low system resources and the system will work in almost all configurations. It has got following features

- Enhanced Security
- Improved Robustness
- High Imperceptibility
- Efficient Embedding and Extraction
- Capacity and Flexibility

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. Three key considerations involved in the feasibility analysis are

- Economic Feasibility
- Technical Feasibility
- Behavioural Feasibility

2.4.1 ECONOMIC FEASIBILITY

In the development of the secured watermarking technology, economic feasibility involves assessing the costs associated with the implementation and ensuring that they align with the available budget. This project leverages open-source software tools and freely available technologies, which minimizes development costs. By avoiding expensive proprietary tools, the project remains financially viable while meeting the necessary security and robustness requirements. The only expenditures involved were for specialized components essential to the cryptographic processes and bit pairs matching. Overall, the project's budget remains manageable and cost-effective, making it a financially feasible solution for securing multimedia content.

2.4.2 TECHNICAL FEASIBILITY

Technical feasibility examines whether the existing infrastructure and resources are sufficient to support the secured watermarking technology. This project was designed to integrate seamlessly with standard computing environments, requiring no extensive hardware upgrades or specialized systems. The system employs Python, along with libraries for image processing and cryptography, all of which are compatible with most technical environments and easy to implement. Given that the system does not demand significant computational power or specialized hardware, it is accessible and practical for a wide range of users. This ensures that the secured watermarking technology can be deployed without putting undue strain on technical resources.

2.4.3 BEHAVIOURAL FEASIBILITY

Behavioural feasibility addresses the user acceptance and the ease with which they can adapt to the secured watermarking technology. For this project, the user interface has been developed with simplicity in mind, ensuring that users can quickly learn and effectively use the system. Training sessions and comprehensive documentation will be provided to build user confidence in embedding and extracting watermarks securely. Additionally, by allowing users to see the clear benefits of protecting multimedia content through watermarking, they are more likely to embrace the technology as an essential tool rather than a complicated addition. Open channels for feedback and constructive criticism further enhance user satisfaction, ensuring that the technology is well-received and readily adopted by its intended users.

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specifies the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

- Processor : Intel i5 or Ryzen 5
- Hard disk : 40GB and Above.
- RAM : 4GB and Above.
- Monitor : 5 inches or above.

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements.

- Operating system : Windwos 11
- Languages : Python 3.7.0
- IDE : VS Code

3.ARCHITECTURE

3.ARCHITECTURE

3.1 PROJECT ARCHITECTURE

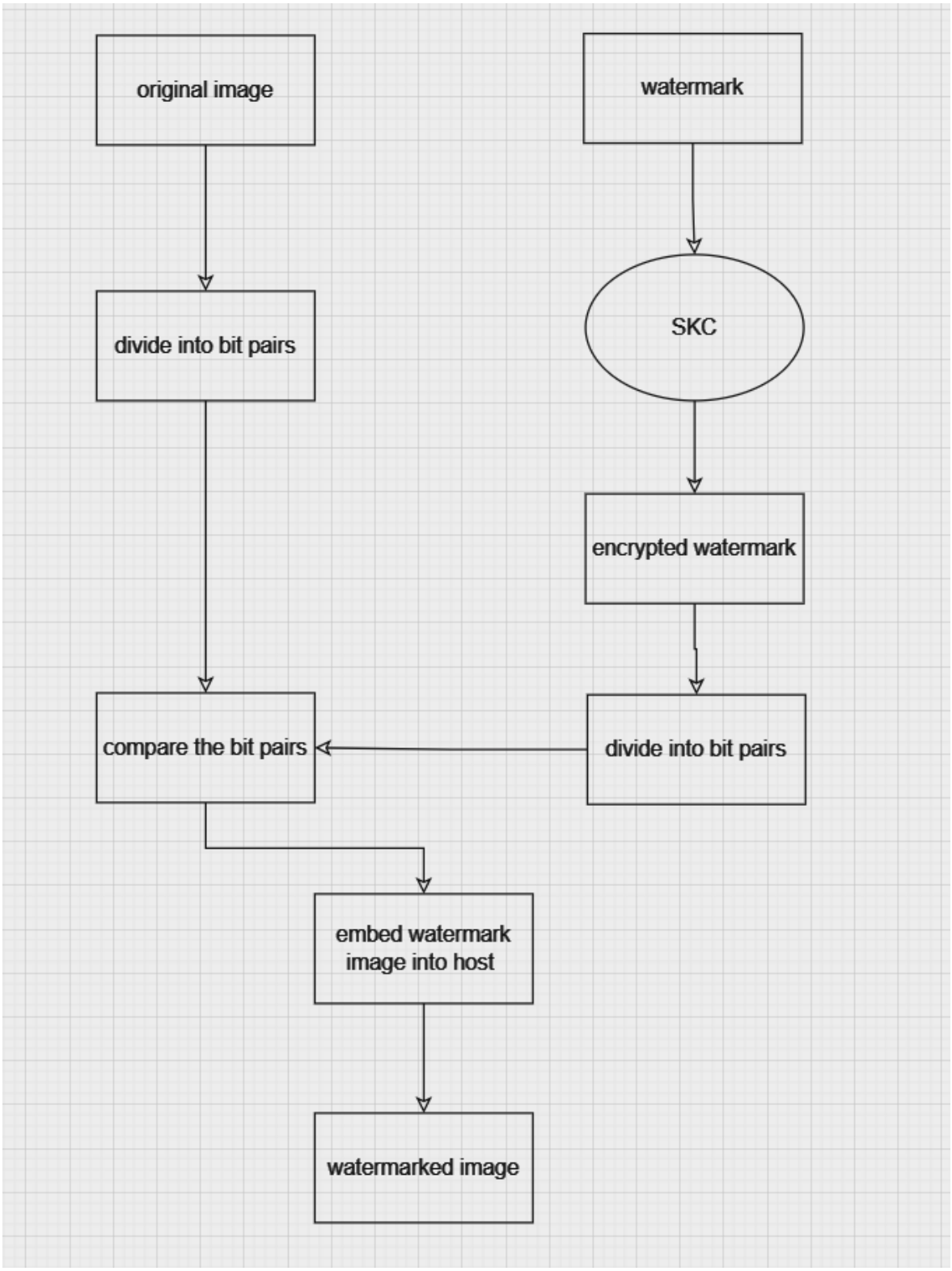


Figure 3.1: Project Architecture embedding watermark using bit pairs matching

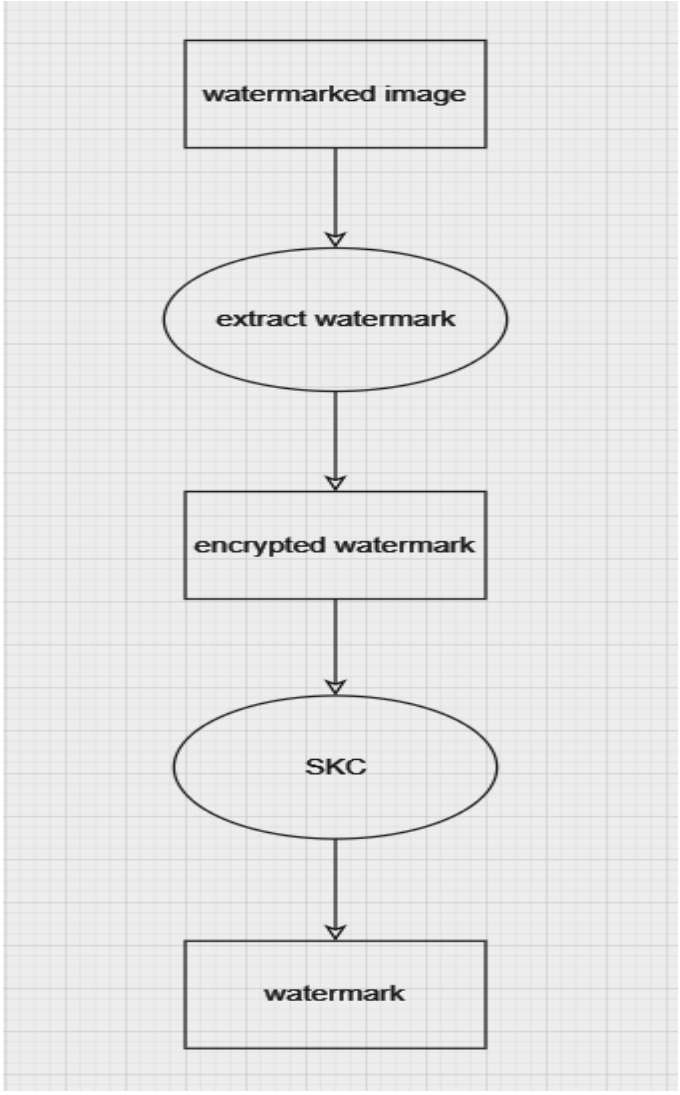


Figure 3.2: Project Architecture extracting watermark from watermarked image

3.2 DESCRIPTION

Encryption: The watermark is first encrypted using a symmetric key cryptographic algorithm to enhance security during transmission.

Bit Pairs Matching: The encrypted watermark and the cover image are processed, forming bit pairs. These pairs are then matched to determine the best location for embedding..

Embedding: The matched bit pairs of the encrypted watermark are embedded into the cover image, replacing the least significant bits of the corresponding pixels.

Extraction: To retrieve the watermark, the LSBs of the watermarked image are extracted, converted to decimal values, and used to reconstruct the encrypted watermark.

Decryption: Finally, the decrypted watermark is obtained using the same symmetric key used in the encryption stage, restoring the original watermark.

3.3 USE CASE DIAGRAM

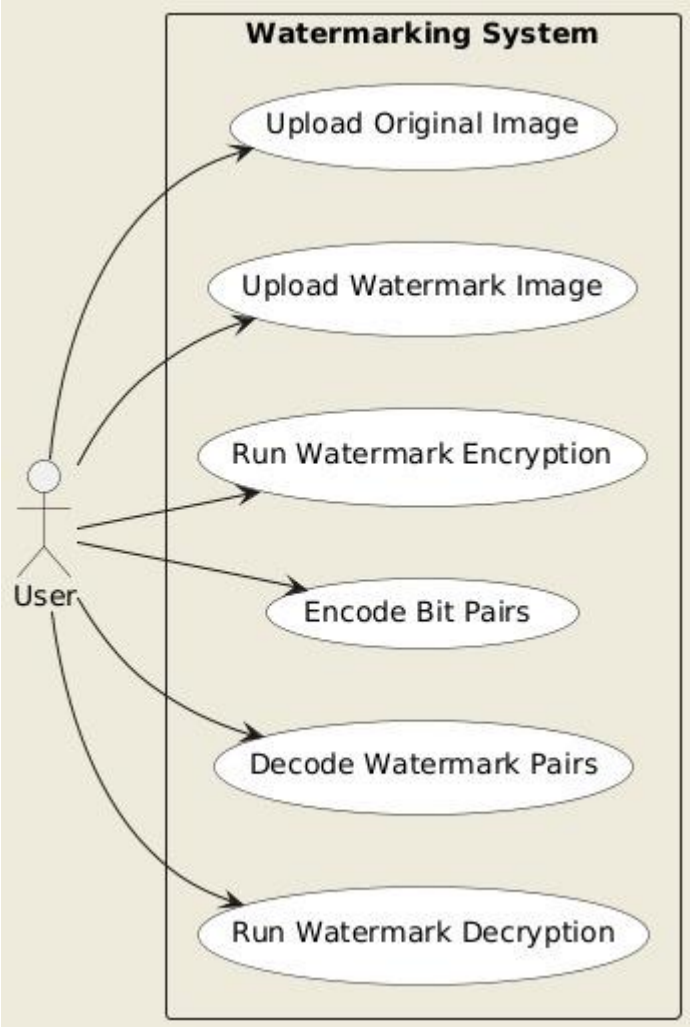


Figure 3.3: Use Case Diagram for secured watermarking system

3.4 CLASS DIAGRAM

Class Diagram is a collection of classes and objects.

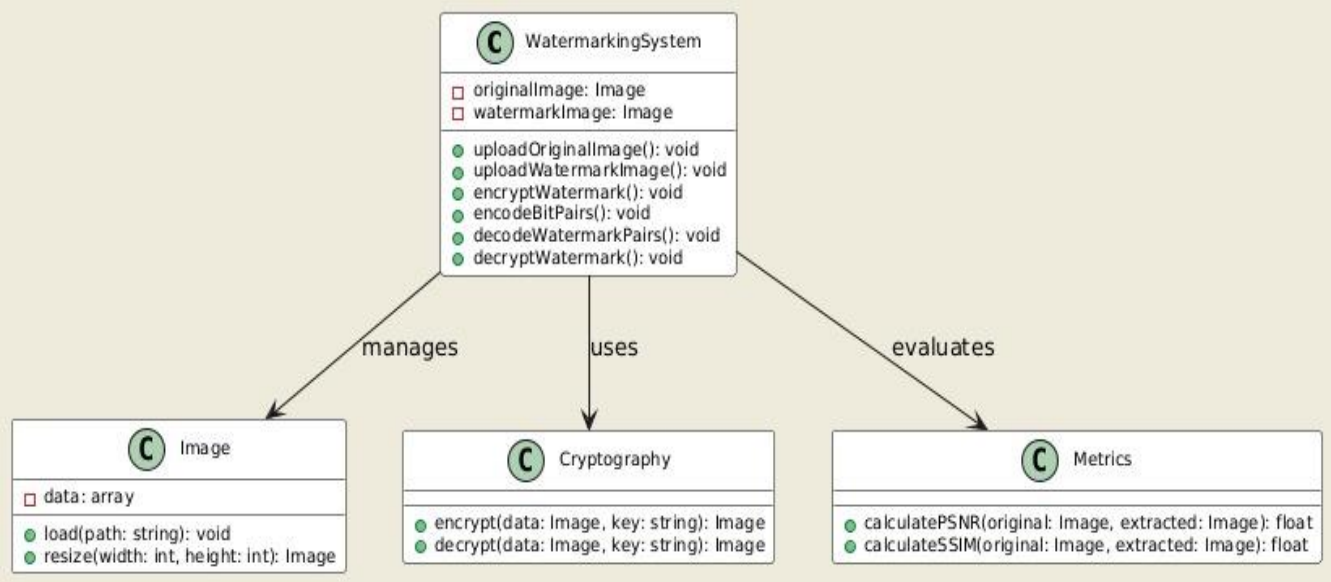


Figure 3.4: Class Diagram for secured watermarking mechanism using cryptography and BPM

3.5 SEQUENCE DIAGRAM

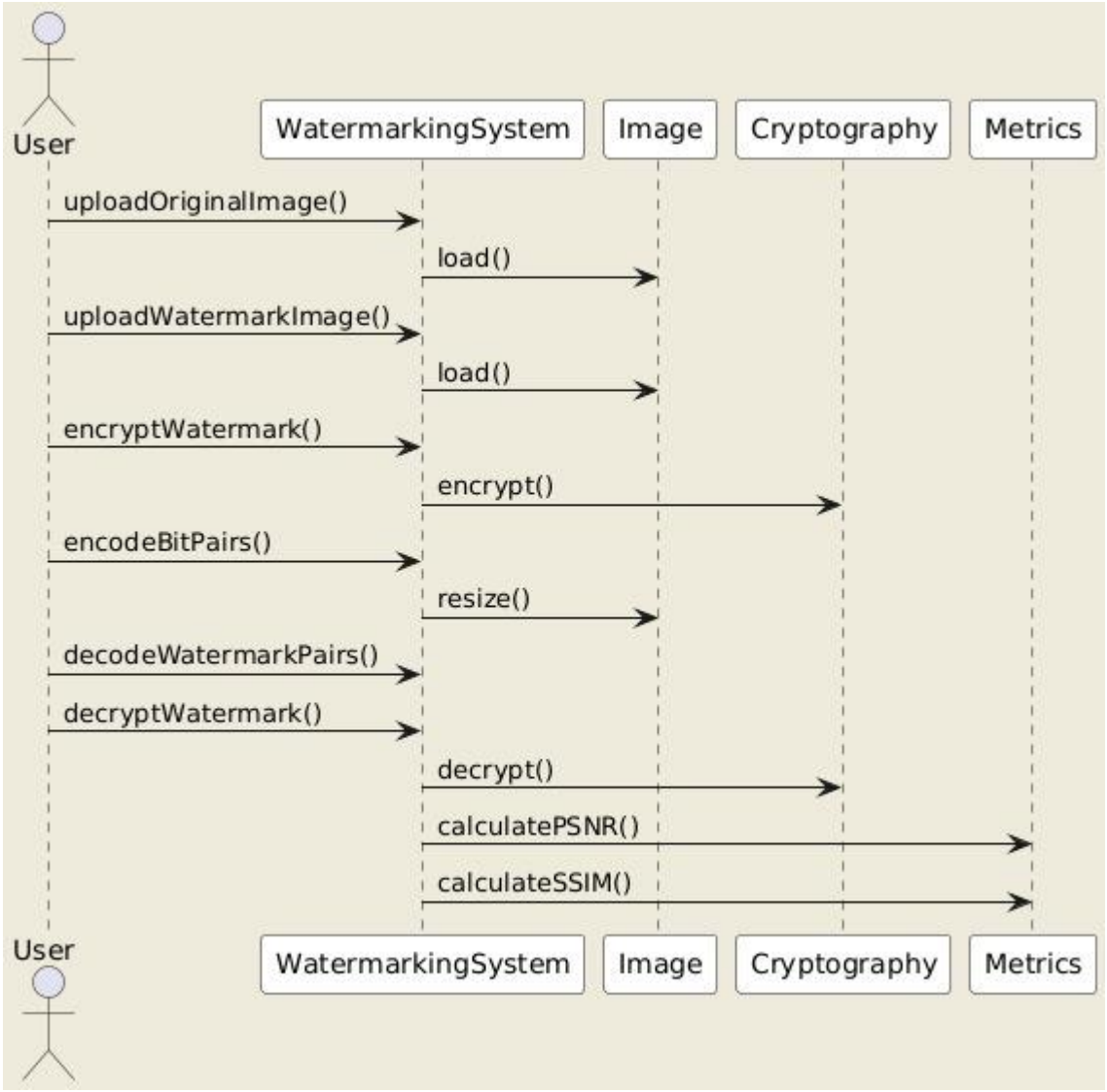


Figure 3.5: Sequence Diagram for secured watermarking mechanism using cryptography and BPM

3.6 ACTIVITY DIAGRAM

It describes about flow of activity states.

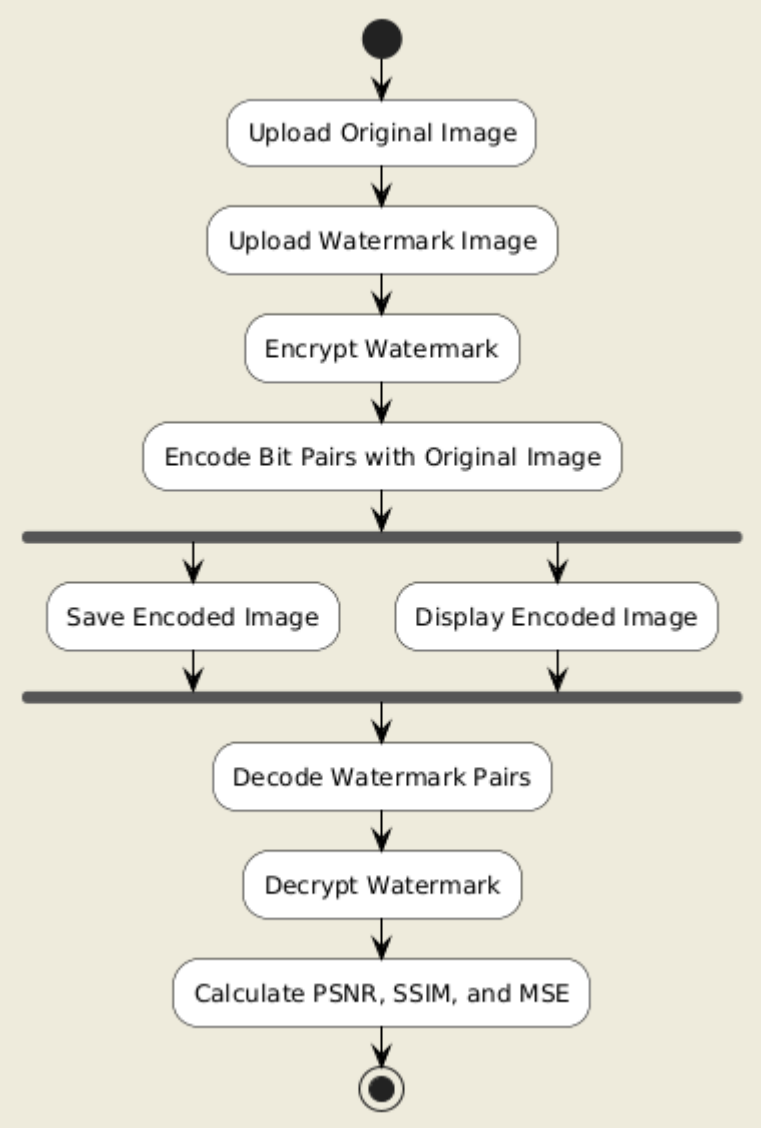


Figure 3.6: Activity Diagram for User to encrypt and decrypt watermarks

4.IMPLEMENTATION

4. IMPLEMENTATION

4.1 SAMPLE CODE

```
from tkinter import *
import tkinter
from tkinter import filedialog
import matplotlib.pyplot as plt
from tkinter.filedialog import askopenfilename
import numpy as np
import cv2
from math import log10, sqrt
import os
from skimage.metrics import structural_similarity as ssim
import pywt

main = tkinter.Tk()
main.title("On the implementation of a secured watermarking mechanism based on cryptography and bit pairs matching")
main.geometry("1200x1200")

global host, watermark

# Ensure model directory exists
if not os.path.exists("model"):
    os.makedirs("model")

# PSNR function
def PSNR(original, compressed):
    mse = np.mean((original - compressed) ** 2)
    if mse == 0:
        return 100
    max_pixel = 255.0
    psnr = 100 - (20 * log10(max_pixel / sqrt(mse)))
    return psnr, mse

# SSIM function
def imageSSIM(normal, embed):
    ssim_value = ssim(normal, embed, data_range=embed.max() - embed.min())
    return ssim_value

# Host image upload
def uploadHost():
    global host
    text.delete('1.0', END)
    host = filedialog.askopenfilename(initialdir="hostImages")
    pathlabel.config(text=f"{host} host image loaded")

# Watermark image upload
def uploadWatermark():
    global watermark
    watermark = filedialog.askopenfilename(initialdir="watermarkImages")
    pathlabel.config(text=f"{watermark} watermark image loaded")
```

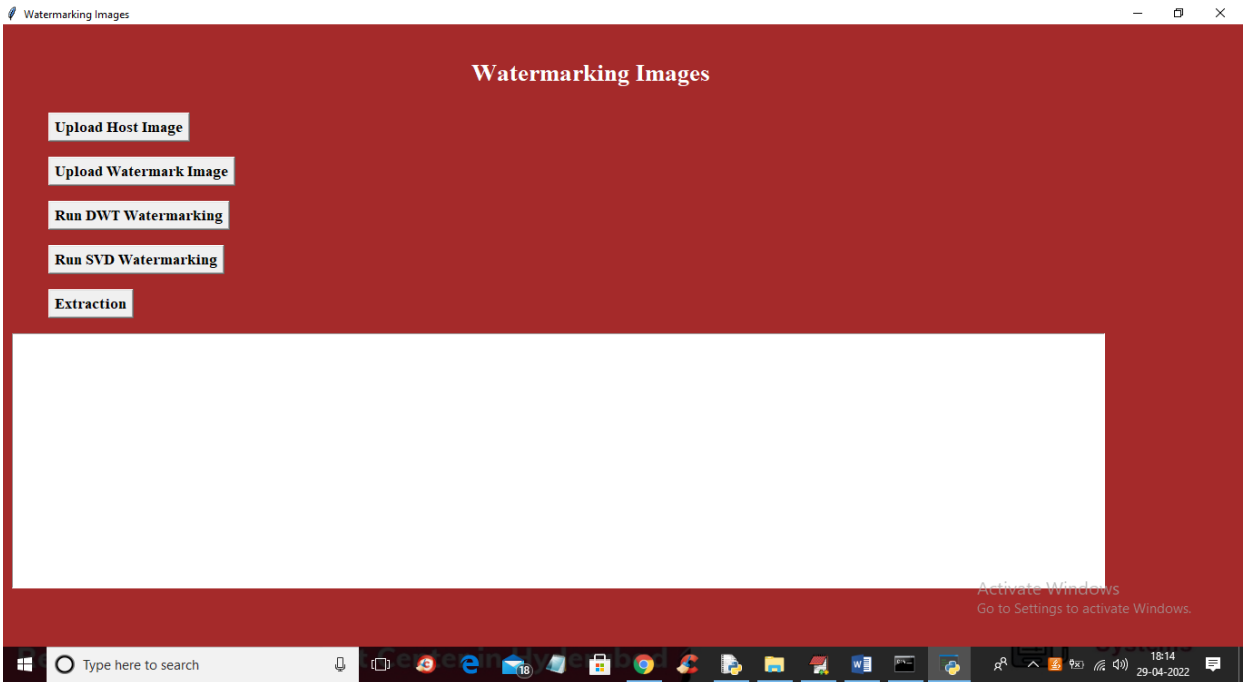
```
# Run DWT for watermark embedding
def runDWT():
    text.delete('1.0', END)
    global host, watermark
    coverImage = cv2.imread(host, 0)
    watermarkImage = cv2.imread(watermark, 0)

    coverImage = cv2.resize(coverImage, (300, 300))
    cv2.imshow('Cover Image', cv2.resize(coverImage, (400, 400)))
    watermarkImage = cv2.resize(watermarkImage, (150, 150))
    cv2.imshow('Watermark Image', cv2.resize(watermarkImage, (400, 400)))

    coverImage = np.float32(coverImage) / 255
    coeffC = pywt.dwt2(coverImage, 'haar')
    cA, (cH, cV, cD) = coeffC
    watermarkImage = np.float32(watermarkImage) / 255

    # Embedding
    coeffW = (0.4 * cA + 0.1 * watermarkImage, (cH, cV, cD))
    watermarkedImage = pywt.idwt2(coeffW, 'haar')
    psnr, mse = PSNR(coverImage, watermarkedImage)
    ssim_value = imageSSIM(coverImage, watermarkedImage)
    text.insert(END, f"DWT PSNR : {psnr}\n")
    text.insert(END, f"DWT MSE   : {mse}\n")
    text.insert(END, f"DWT SSIM : {ssim_value}\n\n")
    text.update_idletasks()

    # Save the watermarked image and coefficients
    name = os.path.basename(host)
    cv2.imwrite(f"OutputImages/{name}", watermarkedImage * 255)
    np.save(f"model/{name}", watermarkedImage)
    np.save(f"model/CA_{name}", cA)
    cv2.imshow('Watermarked Image', cv2.resize(watermarkedImage, (400, 400)))
    cv2.waitKey(0)

# Run extraction
def runExtraction():
    wm = filedialog.askopenfilename(initialdir="OutputImages")
    wm = os.path.basename(wm)
    try:
        img = np.load(f"model/{wm}.npy")
        cA = np.load(f"model/CA_{wm}.npy")
        coeffWM = pywt.dwt2(img, 'haar')
        hA, (hH, hV, hD) = coeffWM

        extracted = (hA - 0.4 * cA) / 0.1
        extracted *= 255
        extracted = np.uint8(extracted)
        extracted = cv2.resize(extracted, (400, 400))
        cv2.imshow('Extracted Image', extracted)
        cv2.waitKey(0)
    except FileNotFoundError:
        text.insert(END, f"File model/{wm}.npy or model/CA_{wm}.npy not found.\n")
        text.update_idletasks()
```

```
# Run SVD (left unchanged as per your original code)
def runSVD():
    coverImage = cv2.imread(host, 0)
    watermarkImage = cv2.imread(watermark, 0)
     cv2.imshow('Cover Image', cv2.resize(coverImage, (400, 400)))
    coverImage = np.double(coverImage)
    watermarkImage = np.double(watermarkImage)

      [m, n] = np.shape(coverImage)

    # SVD of cover image
    ucvr, wcvr, vtcvr = np.linalg.svd(coverImage, full_matrices=1, compute_uv=1)
    Wcvr = np.zeros((m, n), np.uint8)
    Wcvr[:m, :n] = np.diag(wcvr)
    Wcvr = np.double(Wcvr)
    [x, y] = np.shape(watermarkImage)

    # Modifying diagonal component
    for i in range(x):
        for j in range(y):
            Wcvr[i, j] = (Wcvr[i, j] + 0.01 * watermarkImage[i, j]) / 255

    # SVD of modified wcvr
    u, w, v = np.linalg.svd(Wcvr, full_matrices=1, compute_uv=1)

    # Watermarked Image
    S = np.zeros((512, 512), np.uint8)
    S[:m, :n] = np.diag(w)
    S = np.double(S)
    wimg = np.matmul(ucvr, np.matmul(S, vtcvr))
    wimg *= 255
    watermarkedImage = np.zeros(wimg.shape, np.double)
    cv2.normalize(wimg, watermarkedImage, 1.0, 0.0, cv2.NORM_MINMAX)
    psnr, mse = PSNR(coverImage, watermarkedImage)
    ssim_value = imageSSIM(coverImage, watermarkedImage)
    text.insert(END, f"SVD PSNR : {psnr}\n")
    text.insert(END, f"SVD MSE   : {mse}\n")
    text.insert(END, f"SVD SSIM : {ssim_value}\n\n")
    text.update_idletasks()
    cv2.imshow('Watermarked Image', cv2.resize(watermarkedImage, (400, 400)))
    cv2.waitKey(0)

# GUI Layout
font = ('times', 20, 'bold')
title = Label(main, text='On the implementation of a secured watermarking mechanism based on cryptography
and bit pairs Matching')
title.config(bg='brown', fg='white')
title.config(font=font, height=3, width=80)
title.place(x=5, y=5)

font1 = ('times', 13, 'bold')
uploadHost = Button(main, text="Upload Host Image", command=uploadHost)
uploadHost.place(x=50, y=100)
uploadHost.config(font=font1)
```

```
loadwatermark = Button(main, text="Upload Watermark Image", command=uploadWatermark)
loadwatermark.place(x=50, y=150)
loadwatermark.config(font=font1)

pathlabel = Label(main)
pathlabel.config(bg='brown', fg='white')
pathlabel.config(font=font1)
pathlabel.place(x=380, y=100)



dwtButton = Button(main, text="Run Bit Pairs Matching", command=runDWT)
dwtButton.place(x=50, y=200)
dwtButton.config(font=font1)

svdButton = Button(main, text="Embedding the data", command=runSVD)
svdButton.place(x=50, y=250)
svdButton.config(font=font1)

extractionButton = Button(main, text="Extraction", command=runExtraction)
extractionButton.place(x=50, y=300)
extractionButton.config(font=font1)

font1 = ('times', 12, 'bold')
text = Text(main, height=15, width=150)
scroll = Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=10, y=350)
text.config(font=font1)

main.config(bg='brown')
main.mainloop()
```

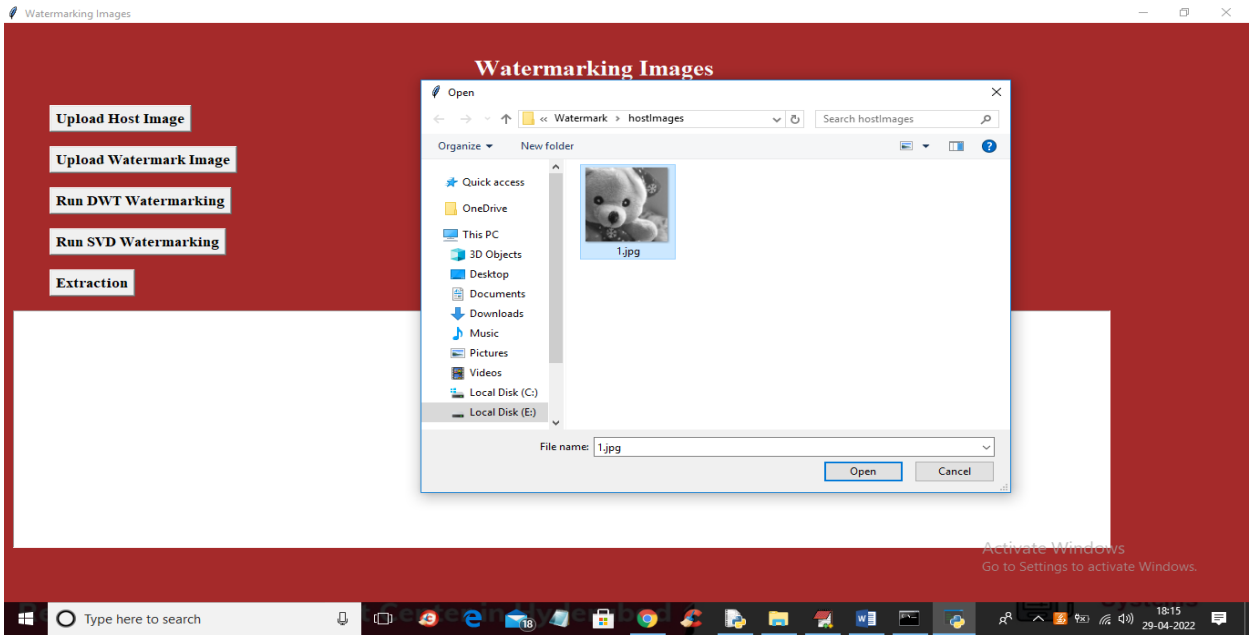
5.SCREENSHOTS

5.1 Start Page



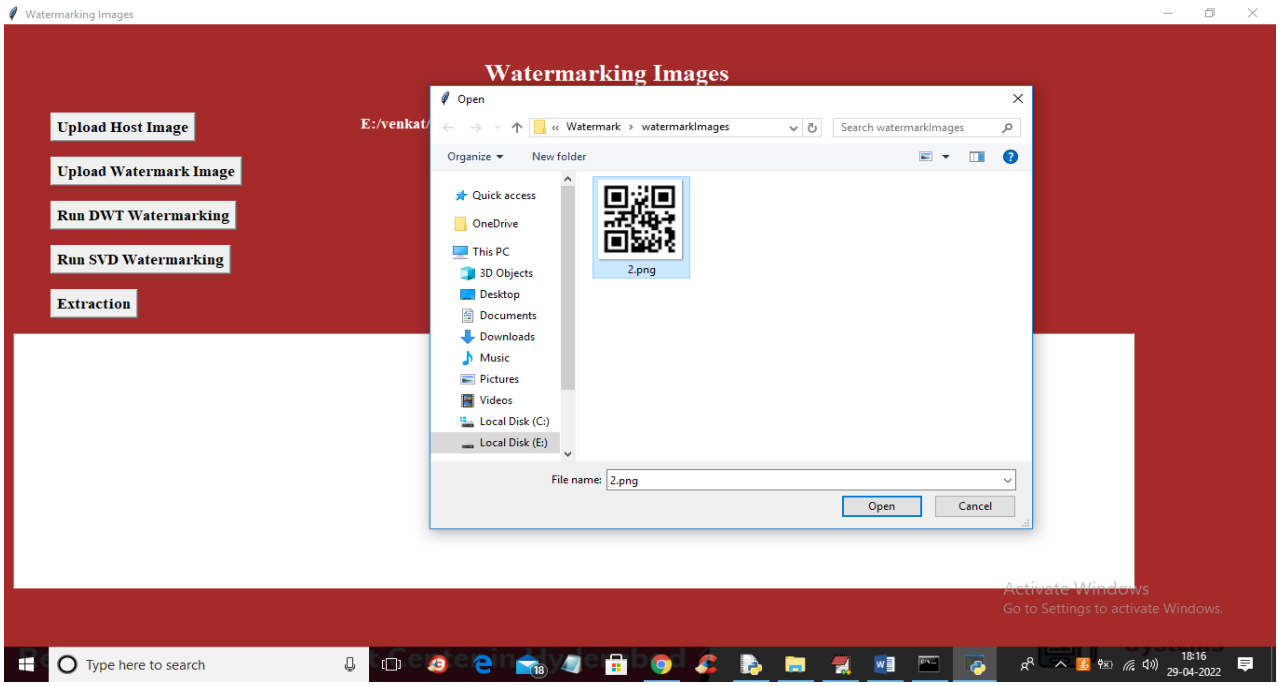
Screenshot 5.1: Start page of the program

5.2 Uploading Host Image



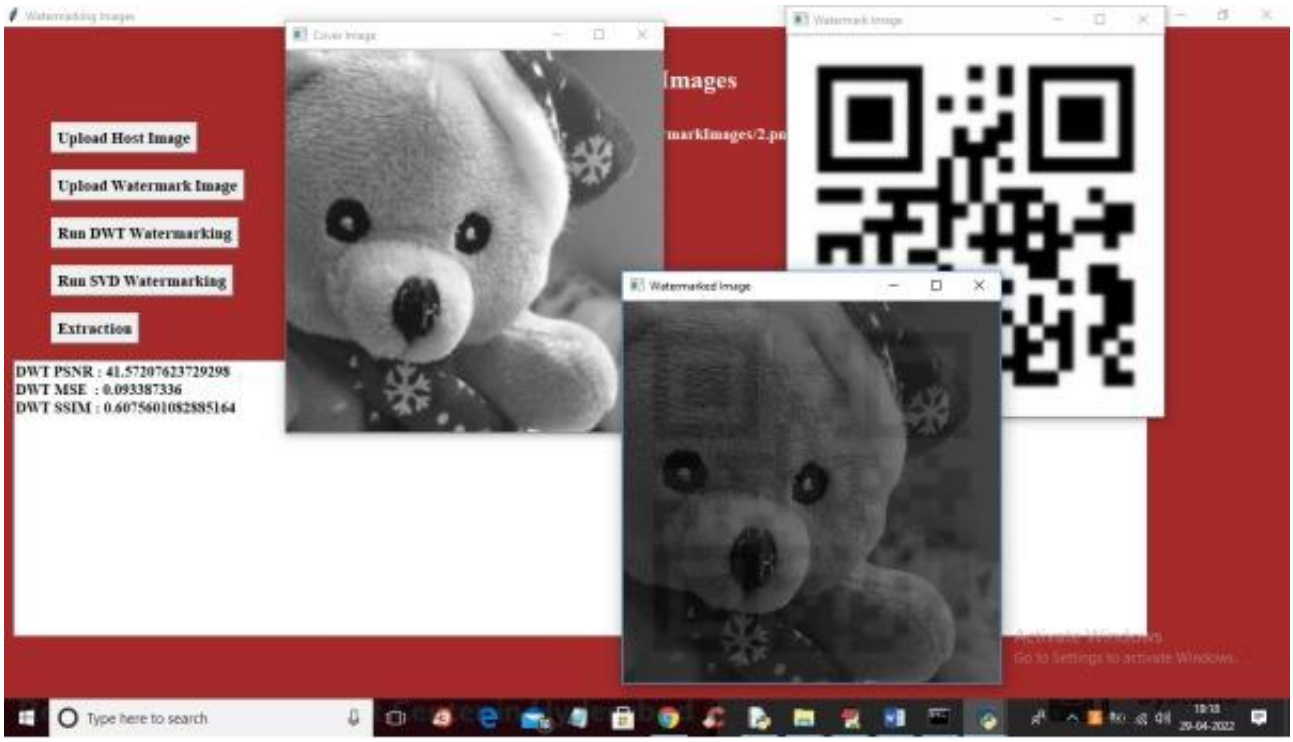
Screenshot 5.2: Interface for uploading host image

5.3 Uploading Watermark Image



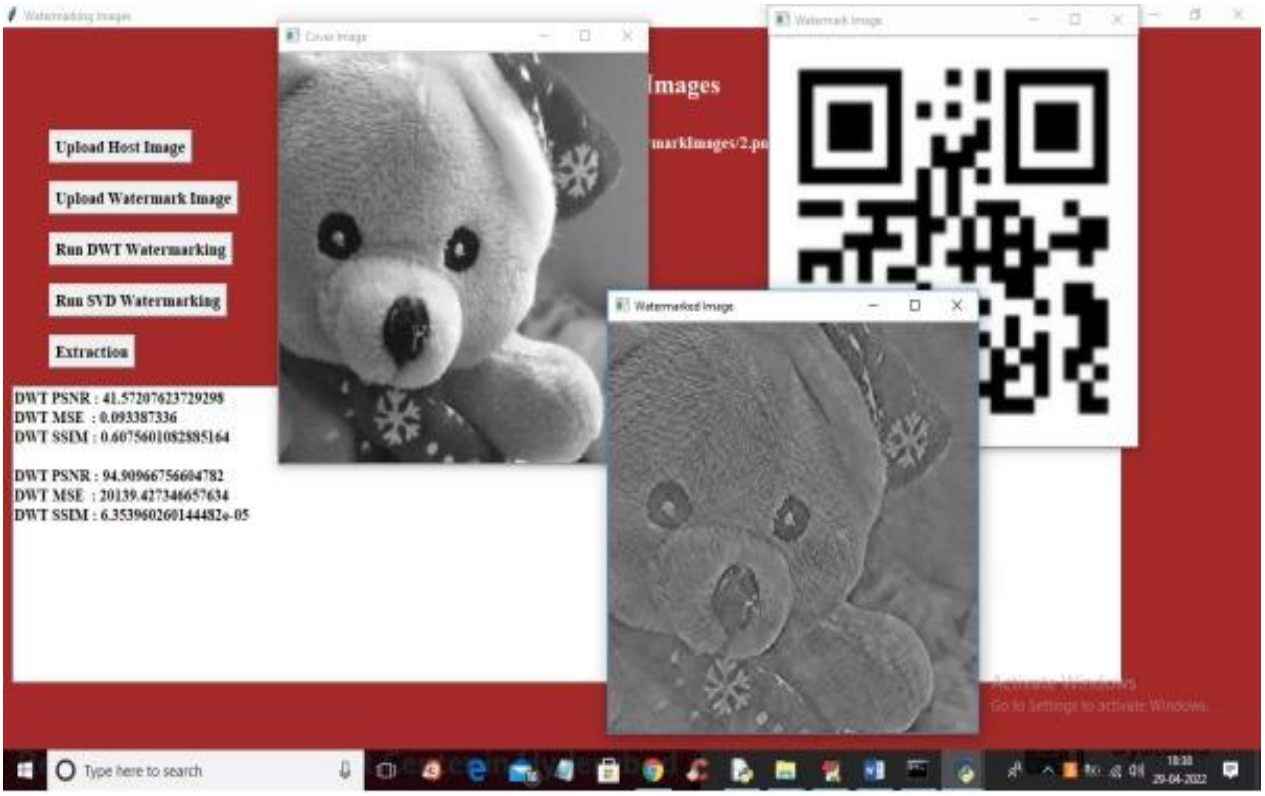
Screenshot 5.3:Interface for uploading Watermark Image

5.4 RESULTS AFTER RUNNNIG THE DWT WATERMARKING



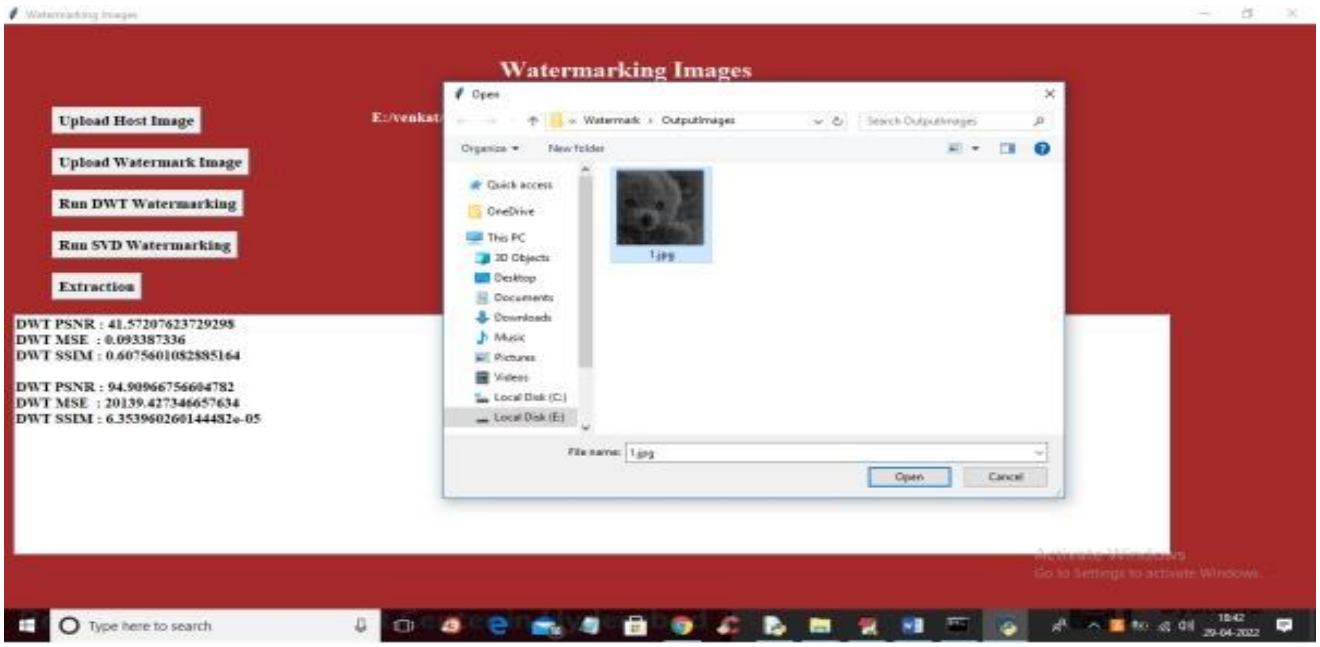
Screenshot 5.4 : Results after running dwt watermarking.

5.5 RESULTS AFTER SVD WATERMARKING



Screenshot 5.5:Results after running svd watermarking

5.6 INTERFACE FOR EXTRACTING WATERMARK



Screenshot 5.6: Interface for Extracted watermark

5.7 EXTRACTED WATERMARK

Screenshot 5.7:Extracted watermark

6.TESTING

6. TESTING

6.1 INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.2 TYPES OF TESTING

6.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes.

6.3 TEST CASES

6.3.1 TEST RESULTS OF IMPERCEPTIBILITY

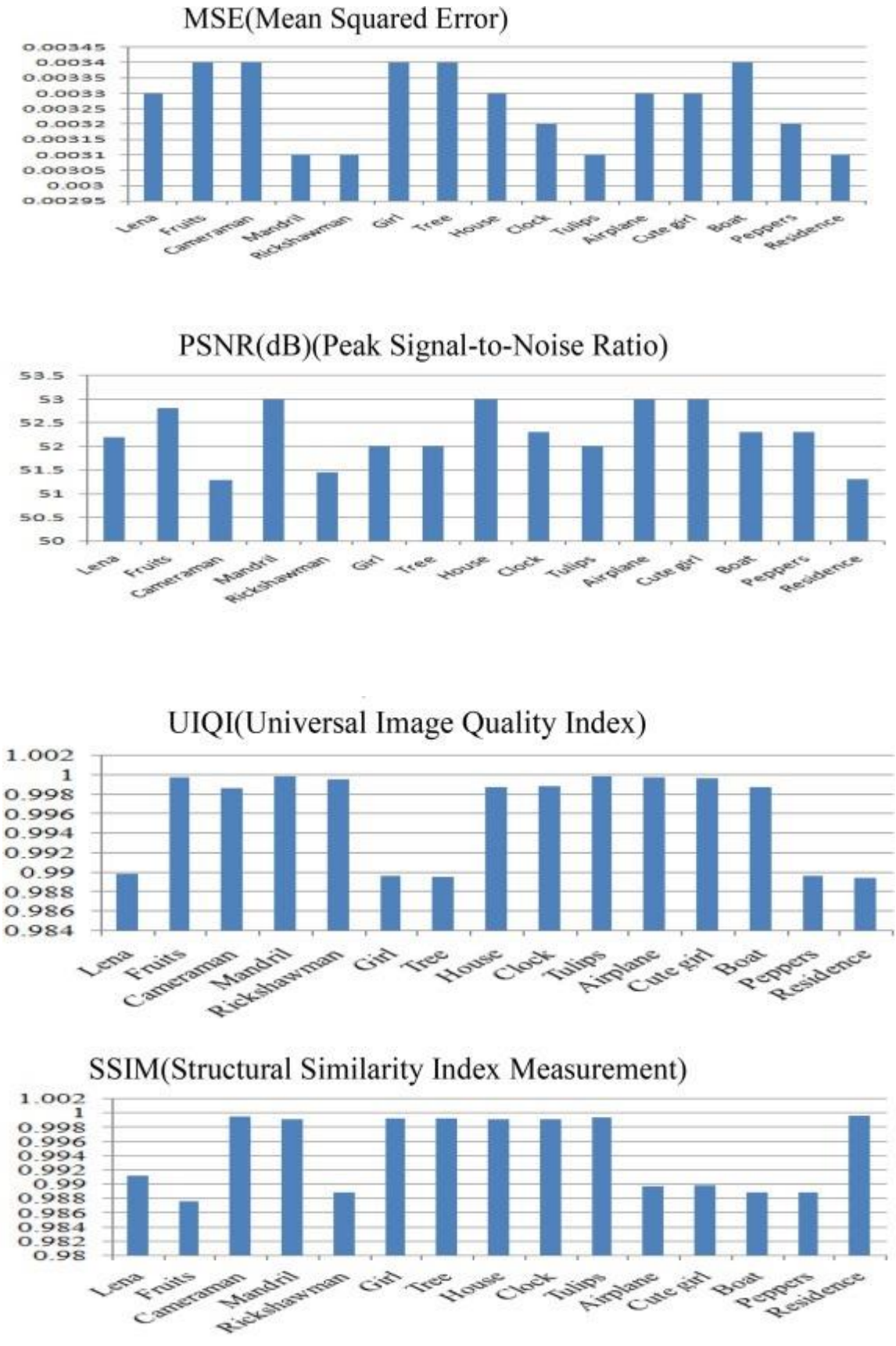


Fig 6.3.1.1-4 Graphs for test results of imperceptibility

6.3.2 TEST RESULTS FOR ROBUSTNESS

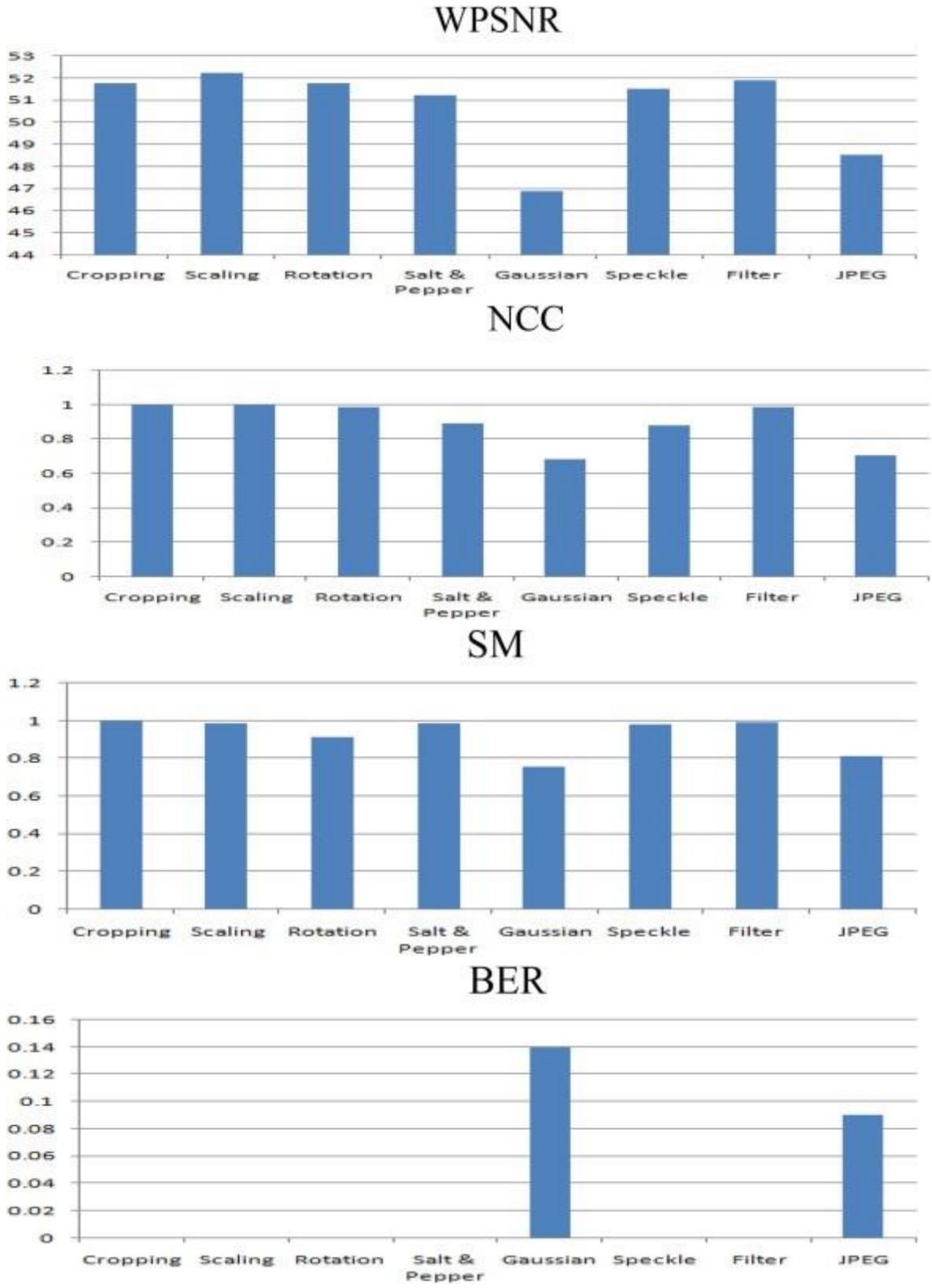


Fig 6.3.2.1-4 Graphs for test results of robustness

7.CONCLUSION & FUTURE SCOPE

7. CONCLUSION & FUTURE SCOPE

7.1 PROJECT CONCLUSION

Watermarks performance against different attacks.This project successfully implements a secured watermarking mechanism based on cryptography and bit pairs matching techniques. By embedding watermark data into host images using wavelet transforms and bit-level manipulation, the system ensures that the watermark is imperceptible, while still being recoverable through cryptographic methods. Additionally, the project evaluates the performance of watermark embedding and extraction using metrics such as PSNR, SSIM, and MSE, which show how well the original and watermarked images are preserved in terms of quality and similarity.

The use of symmetric cryptography in this project adds a layer of security to the watermarking process, ensuring that only authorized users can decode the embedded watermark. The system demonstrates effective watermark extraction without significantly degrading the visual quality of the host image, making it suitable for secure data transmission, copyright protection, and media authentication.

7.2 FUTURE SCOPE

The future scope of this project includes enhancing robustness against common attacks like noise addition, scaling, and compression by incorporating hybrid techniques (e.g., DWT with DCT or SVD). Further, integrating asymmetric cryptographic algorithms can strengthen security, while dynamic watermarking can improve invisibility and resilience. Expanding to video watermarking, implementing machine learning for enhanced detection, and developing real-time watermarking for live streams offer exciting extensions. Lastly, blockchain integration can provide immutable proof of ownership and content authenticity tracking, adding value to digital rights management systems.

8.BIBLIOGRAPHY

8. BIBLIOGRAPHY

8.1 REFERENCES

1.Al-Otaibi, N.A., Gutub, A.A.A., 2014. Flexible stego-system for hiding text in images of personal computers based on user security priority. In: Proceedings of 2014 International Conference on Advanced Engineering Technologies (AET-2014), Dubai UAE, pp. 250–256.

2.Altaibi, N.A., Gutub, A.A., Khan, E.A., 2015. Stego-system for hiding text in images of personal computers. In: The 12th Learning and Technology Conference: Wearable Tech/Wearable Learning, Effat University, Jeddah, Kingdom of Saudi Arabia.

3.Gutab, A.A.A., Ghouti, L., 2007. Utilizing extension character 'Kashida' with pointed letters for arabic text digital watermarking. In: International Conference on Security and Cryptography (SECRYPT), Barcelona, Spain.

4.Gutub, A., Al-Qahtani, A., Tabakh, A., 2009. Triple-A: secure RGB image steganography based on randomization. In: The 7th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA-2009), Rabat, Morocco, pp. 400–403

5.Hannigan, B.T., Reed, A., Bradley, B., 2001. Digital watermarking using improved human visual system model. In: Ping Wah Wong, Edward J. Delp (Eds.), Proc. SPIE. 4314, 468-474, Security and Watermarking of Multimedia Contents III.

6. Hempstalk, K., 2006. Hiding behind corners: using edges in images for better steganography. In: Proceedings of the Computing Women's Congress, Hamilton, New Zealand, pp. 11–19

7.Khan, F., Gutub, A.A.A., 2007. Message concealment techniques using image based steganography. In: The 4th IEEE GCC Conference and Exhibition, Gulf International Convention Centre, Manamah, Bahrain.

8.Kutter, M., Hartung, F., 1999. Image watermarking techniques. In: Proceedings of the IEEE, Special Issue on Identification and Protection of Multimedia Information.

9.Mahjabin, T., Hossain, S., Haque, M., 2012. A block based data hiding method in images using pixel value differencing and LSB substitution method. In: Proc. International Conference on Computer and Information Technology (ICCIT). Chittagong. Bangladesh, pp. 168–172.

8.2 WEBSITES

[1] https://en.wikipedia.org/wiki/Digital_watermarking#Classification

[2] https://en.wikipedia.org/wiki/Cryptography#Modern_cryptography

[3] https://en.wikipedia.org/wiki/Bit_pairing#References

8.3 GITHUB RESPOSITORY LINK

https://github.com/sakethRreddy/On-the-implementation-of-a-secured-watermarking-mechanism-based-on-cryptography-and-bit-pairs