Abstract:

Digital data can be easily copied, modified and forgeries be created by anyone having a computer. Most prone to such malicious attacks are the digital images published in the Internet. Digital Watermarking can be used as a tool for discovering unauthorized data reuse and also for copyright protection. Digital Watermarking is the technique of embedding some identification information known as watermark into the digital data by its owner. On embedding or data hiding a watermarked data is generated. Large numbers of watermarking schemes are currently available. An acceptable Watermarking must possess certain qualities as robustness and imperceptibility. The recent research and developments in the field of Watermarking is reviewed and is subjected to a detailed study in this paper. The current status and issues are then discussed to give direction to future works

WATERMARK FOR DIGITAL IMAGES

1. INTRODUCTION:

The recent growth of networked multimedia systems has caused problems relative to the protection of intcllectual property rights. This is particularly true for image and video data. The types of protection systems involve the use of both encryption and authentication techniques. In this paper we describe a form of authentication known as a watermark. These digital watermarks also offer forgery detection. Several watermarking techniques have been proposed. One uses a checksum on the image data which is embedded in the least significant bits of certain pixels [l]. Others add a maximal length linear shift register sequence to the pixel data and identify the watermark by computing the spatial crosscorrelation function of the sequence and the watermarked image [2]. Watermarks can be image dependent, using independent visual channels [3], or be generated by modulating JPEG coefficients [4]. These watermarks are designed to be invisible, or to blend in with natural camera or scanner noise. Visible watermarks also exist; IBM has developed a proprietary visible watermark to protect images that are part of the digital Vatican library project [5]. In this paper we present a watermark which is a two-dimensional extension of [2]. We describe a forgery detection scheme with a new approach to robustness. The watermark's robustness to mean and median filtering is investigated. We then introduce a second watermark that is robust relative to PEG compression.

1.1 Objective of the project:

The growth of networked multimedia systems has magnified the need for image copyright protection. One approach used to address this problem is to add an invisible structure to an image that can be used to seal or mark it. These smctures are known as digiral watermarks. In this paper we describe two techniques for the invisible marking of images. We analyze the robustness of the watermarks with respect to linear and nonlinear fillering, and JPEG compression. The results show that our watermarks detect all but the most minute changes to the image

2. LITERATURE SURVEY:

“A Watermark Algorithm of Content Authentication for Network Transmission Image Correcting Errors in Terms of Pixels”

This paper proposes a watermark algorithm for image content authentication. It can correct errors in terms of pixels for image which transmits via media network and be tampered by noise or others hackers. There are two characteristics: first, aiming at defects of poor safety transferring on networks and the complicated calculation which are exiting in image encryption algorithm based on chaotic sequences, a new image chaotic encryption algorithm based on sampling without replacement is put forward. Second, the application of combination mapping technique supplies a gap which is exiting in image content authentication watermark. Using the authentication, digital image can be corrected in terms of pixels, and with the minimum embedded capacity and the maximum SNR. The image is divided into blocks by 8 × 8, image content authentication and correcting errors is done on blocks. In the scheme, chaotic encryption not only ensures the safety on network transferring, but also advances correcting rate which can make sure the usage of images transferring on network, at the same time it guarantees the validity of network transmission image.

"A digital watermark,"

The paper discusses the feasibility of coding an "undetectable" digital water mark on a standard 512/spl times/512 intensity image with an 8 bit gray scale. The watermark is capable of carrying such information as authentication or authorisation codes, or a legend essential for image interpretation. This capability is envisaged to find application in image tagging, copyright enforcement, counterfeit protection, and controlled access. Two methods of implementation are discussed. The first is based on bit plane manipulation of the LSB, which offers easy and rapid decoding. The second method utilises linear addition of the water mark to the image data, and is more difficult to decode, offering inherent security. This linearity property also allows some image processing, such as averaging, to take place on the image, without corrupting the water mark beyond recovery. Either method is potentially compatible with JPEG and MPEG processing.

"Watermarking digital images for copyright protection,"A watermark is all invisible mark placed on an image that can only be detected when the image is compared with the original. This mark is designed to identify boththesource of a document as well as its intended recipient. This paper discusses various techniques for embedding such marks in grey scale and colour digital images. It begins specifying the requirements that effective

image watermarking scheme must possess, followed by a review of current and novel techniques based on image transforms

“Digital Watermarking Techniques and its Application towards Digital Halal Certificate: A Survey”
The ease of alterations, distributions and duplications of a digital data makes the multimedia security to be crucial. Digital image watermarking is used in the multimedia security field in order to secure the ownership of a multimedia content from being forged by the illegal distribution and duplication. This survey paper upholds the techniques and algorithms that is used in digital image watermarking. In brief, digital image watermarking will secure the secret information by embedding it in the original data in order to defend the copyrights and also the ownership of the original multimedia data. Spatial domain or on the medium of the wavelets are types of techniques in digital image watermarking. The spatial domains works on the pixels of the image while the frequency domains process the transform coefficient of multimedia data. Each types of techniques represents different types of implementation. In this survey paper, the advantages and disadvantages of both techniques will be explained further as it is the most crucial part for this title. The watermarking architecture and the types of attacks towards the watermarked media is also further explained in this paper. This paper also includes the previous study regarding the digital watermarking towards the Malaysian Halal Certificate using a different methods and algorithms such as Spread Spectrum. Several previous papers has been referred to in the completion of this review paper to ensure the stability of this research.

“Multimedia watermarking techniques”
Multimedia watermarking technology has evolved very quickly during the last few years. A digital watermark is information that is imperceptibly and robustly embedded in the host data such that it cannot be removed. A watermark typically contains information about the origin, status, or recipient of the host data. In this tutorial paper, the requirements and applications for watermarking are reviewed. Applications include copyright protection, data monitoring, and data tracking. The basic concepts of watermarking systems are outlined and illustrated with proposed watermarking methods for images, video, audio, text documents, and other media. Robustness and security aspects are discussed in detail. Finally, a few remarks are made about the state of the art and possible future developments in watermarking technology.

“Digital watermarking techniques for security applications”
Nowadays, the success of internet technology, made our life very much easy and convenient. But the major problem is to secure the data from duplication and unauthorized use. So the digital watermarking is used. With this technology, we embed the secret information into the actual information for protecting it from unauthorized use. By using this technique only authorized user can access the data. It may be classified into two domains that are spatial domain and frequency domain. In this paper, we have briefly discussed about these technologies and their pros and cons.

“A review of image watermarking”
Editing, reproduction and distribution of the digital multimedia are becoming extremely easier and faster with the existence of the Internet and the availability of pervasive and powerful multimedia tools. However, these advances have their drawbacks as well, for example unauthorized tampering of images. Digital watermarking has emerged as a possible method to tackle these issues. A digital watermark is created by inserting a digital signal, or pattern within multimedia content. This embedded information can be used to determine whether the host data are being tampered with or not. This paper presents a review of watermarking where watermarking processes are described briefly and the appropriate properties for different applications are discussed.

“A Brief Review on: Implementation of Digital Watermarking for Color Image using DWT Method”
This paper describes a watermarking scheme for digital color images consisting of invisible watermarks. This scheme involves the embedding of the dual watermarks i.e, robust & fragile. By using these two watermarks for same image makes the scheme more secure. The first watermark is obtained by using DWT in YCbCr space, the second one i.e fragile watermark is used for improving least significant bits (LSB). Thus without differentiating with input image, it could be verified blindly for authentication. Therefore by using robust and fragile watermarking techniques, the original images could be protected.

3. SYSTEM ANALYSIS

3.1 Existing System

A watermark is like a signature. Not to be taken literally, a watermark is a label located somewhere within an image that identifies the artist that created it or provides evidence of authenticity. In some cases, this is a name, like signing a painting, and in others a website address or an Internet alias. In most cases watermarks are placed on images to avoid unauthorized copying or distribution without properly attributing the creator. As such, watermarks are generally easy to find. However, to prevent easy erasure, some artists create detailed, concealed watermarks to better protect their images.

Disadvantages of Existing System:

- Less Accuracy
- Time taking process

3.2 Proposed System

The recent growth of networked multimedia systems has caused problems relative to the protection of intcllectual property rights. This is particularly true for image and video data. The types of protection systems involve the use of both encryption and authentication techniques. In this paper we describe a form of authentication known as a watermark. These digital watermarks also offer forgery detection. Several watermarking techniques have been proposed. One uses a checksum on the image data which is embedded in the least significant bits of certain pixels. Others add a maximal length linear shift register sequence to the pixel data and identify the watermark by computing the spatial cross correlation function of the sequence and the watermarked image. Watermarks can be image dependent, using independent visual channels, or be generated by modulating JPEG coefficients. These watermarks are designed to be invisible, or to blend in with natural camera or scanner noise. Visible watermarks also exist; IBM has developed a proprietary

Advantages of Proposed System:

- High Accuracy.
- Time consumption is less

Modules Information:

To implement this project author has designed following modules

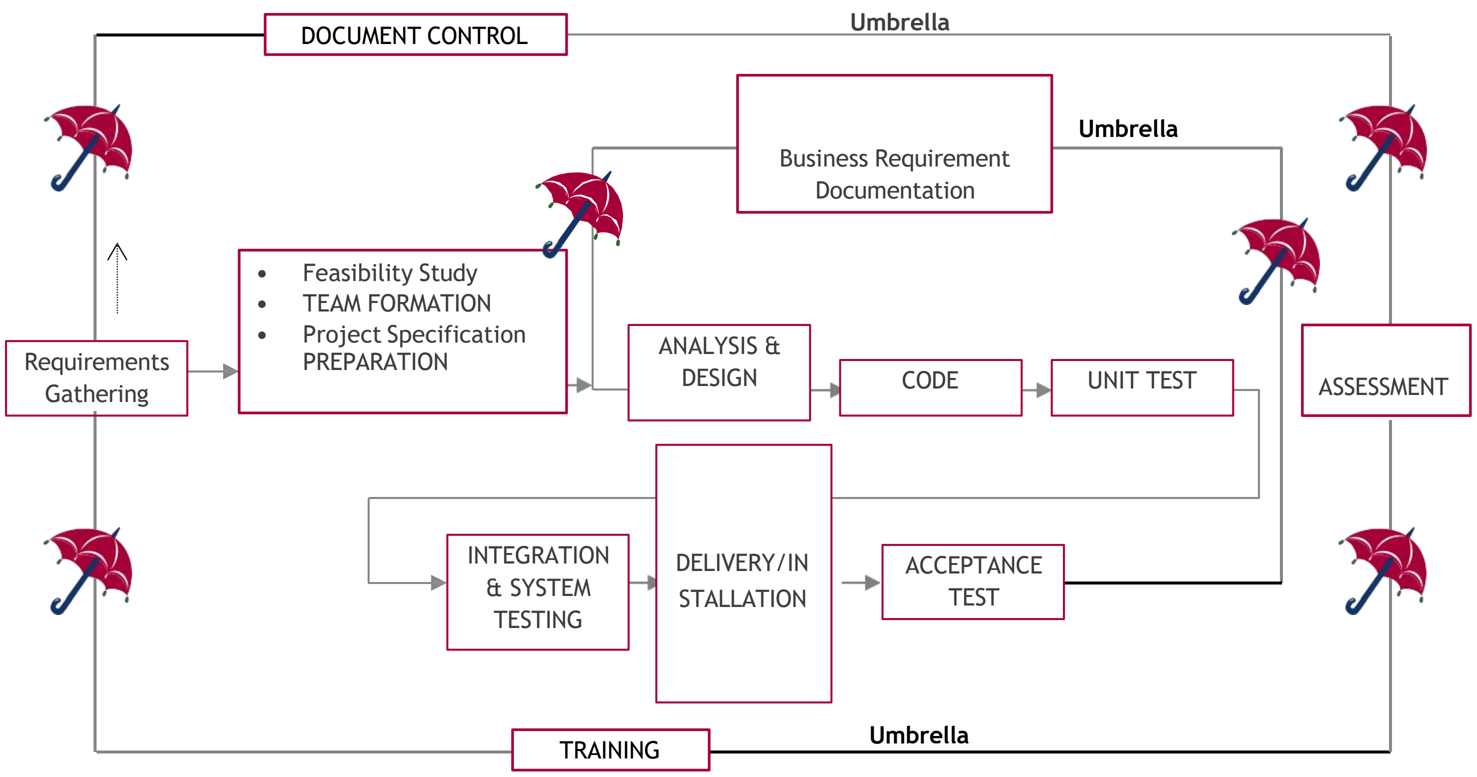
1) upload host image: mage hosting allows people to upload pictures to a specific website. The image hosting website will then store the pictures on its server. It gives different codes for other people to see that picture. With an image hosting site, you can embed images into websites and blogs, making it simple to share your photos.
2) watermark image: Watermarks are used to protect images and visual files from being stolen and used or altered without the owner's permission

3) run DWT water marking: A DWT is applied to the medical image to obtain different frequency sub-bands. Then a DCT is applied to the HH sub-band. An SVD is then applied to the obtained DCT coefficients. Finally, the watermark is integrated into the singular values obtained from the HH high frequency sub-band of the medical image
4) run SVD watermarking: SVD watermarking scheme, which successfully embeds watermarks into images imperceptible way . SVD method can transform matrix A into product USV . Watermarking, is the process of embedding data into a multimedia cover , and can be used primarily for copyright protection and other purposes.
5) extraction: Extraction is the action of removing something. For example, when the dentist yanks out your rotten tooth, the extraction is complete! In addition to this wince-inducing meaning, the noun extraction is the process of separating out something from a chemical mixture or compound.

.

3.3. PROCESS MODEL USED WITH JUSTIFICATION

SDLC (Umbrella Model):



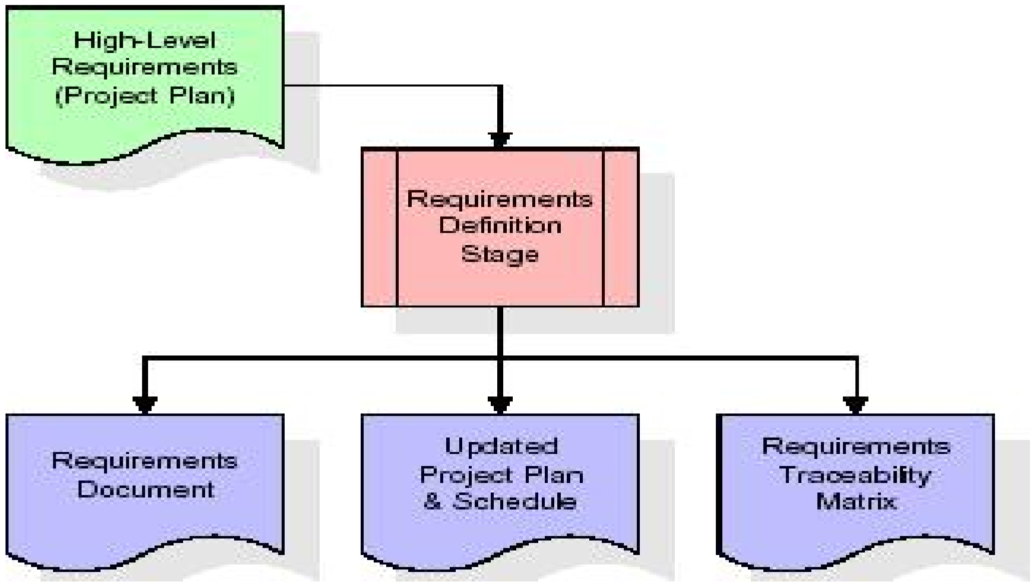
SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

Stages in SDLC:

- Requirement Gathering
- Analysis
- Designing
- Coding
- Testing
- Maintenance

Requirements Gathering stage:

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.


High-Level Requirements (Project Plan)
Requirements Definition Stage
Requirements Document
Updated Project Plan & Schedule
Requirements Traceability Matrix

These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

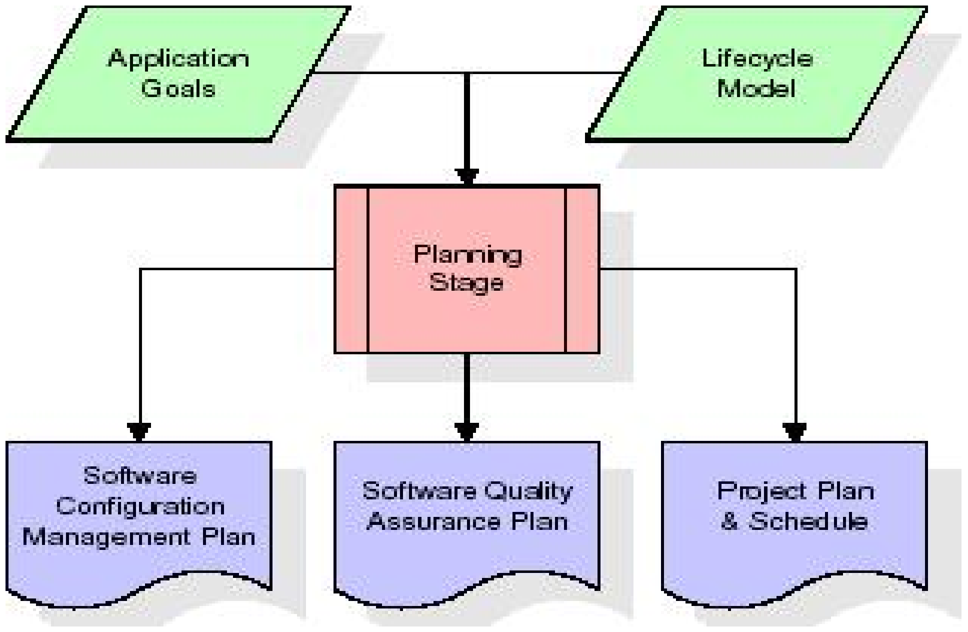
In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

- Feasibility study is all about identification of problems in a project.
- No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
- Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

Analysis Stage:

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.

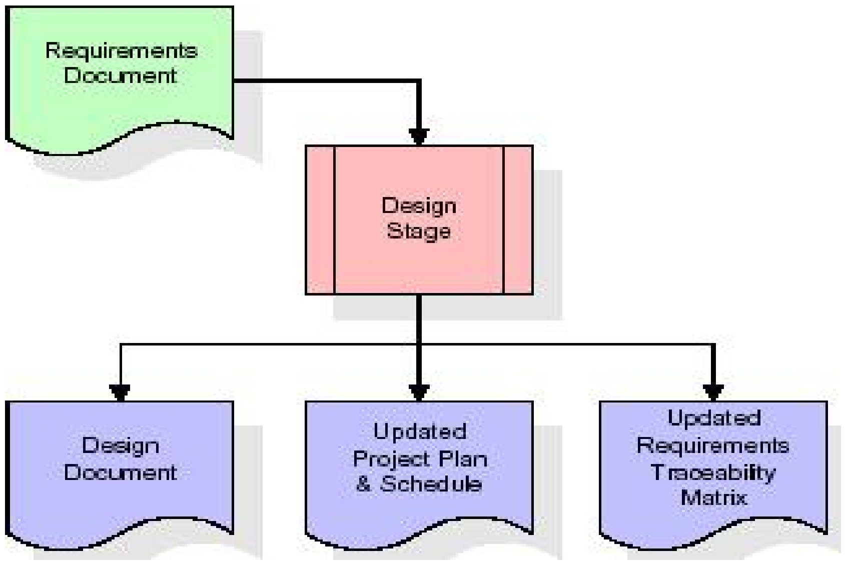

Application Goals
Lifecycle Model
Planning Stage
Software Configuration Management Plan
Software Quality Assurance Plan
Project Plan & Schedule

The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and

schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

Designing Stage:

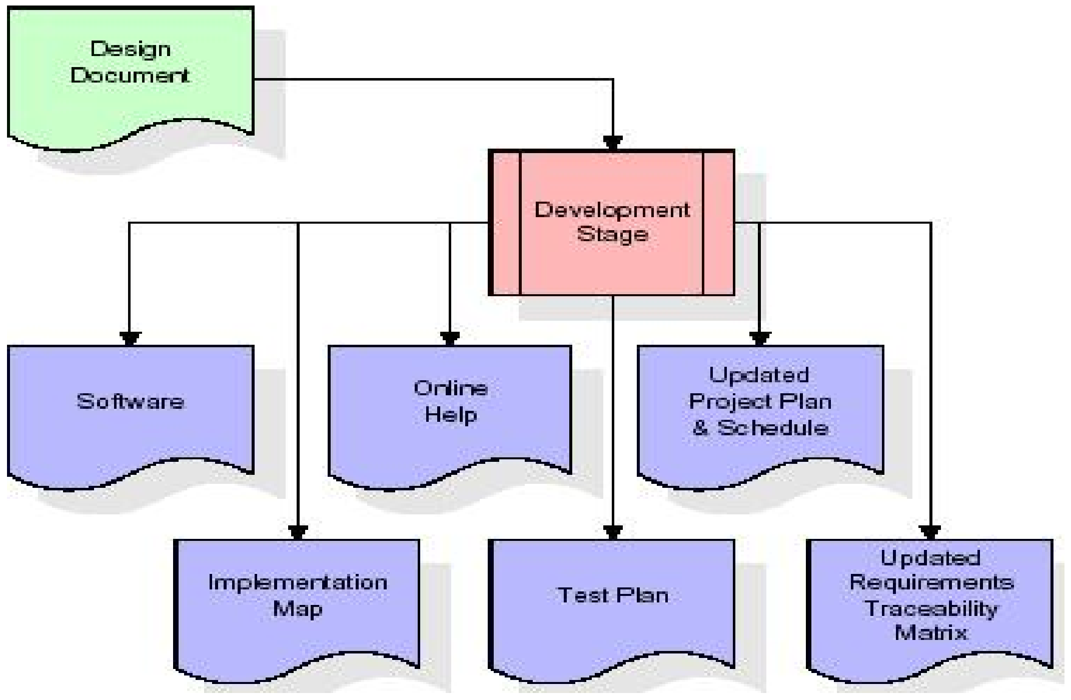
The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.


Requirements Document
Design Stage
Design Document
Updated Project Plan & Schedule
Updated Requirements Traceabilty Matrix

When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

Development (Coding) Stage:

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, and data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.

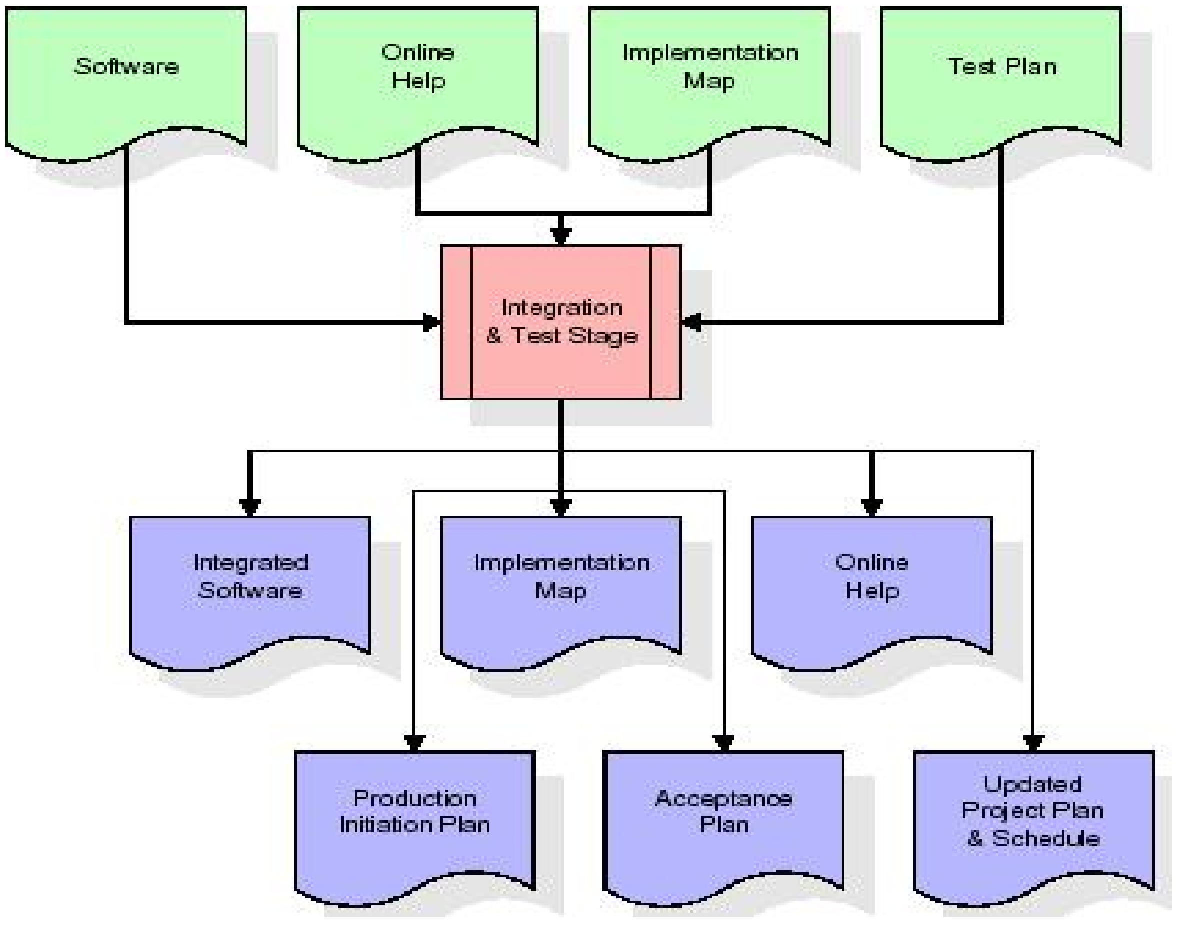


The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

Integration & Test Stage:

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and

complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.

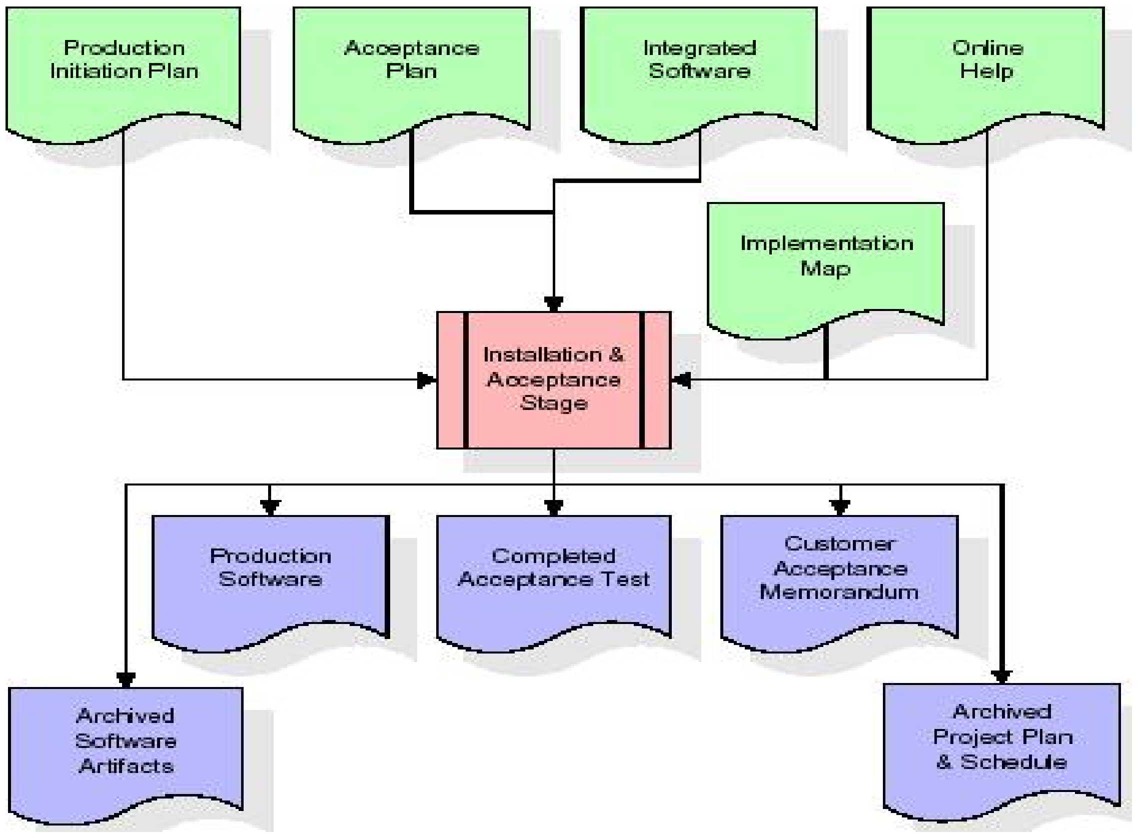


The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

♦ **Installation & Acceptance Test:**

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loa ded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.


Production Initiation Plan
Acceptance Plan
Integrated Software
Online Help
Implementation Map
Installation & Acceptance Stage
Production Software
Completed Acceptance Test
Customer Acceptance Memorandum
Archived Software Artifacts
Archived Project Plan & Schedule

The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labor data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

Maintenance:

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category. For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks).

3.4. Software Requirement Specification

3.4.1. Overall Description

A Software Requirements Specification (SRS) – a requirements specification for a software system is a complete description of the behavior of a system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. Nonfunctional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A business analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the systems development lifecycle domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

- Business requirements describe in business terms what must be delivered or accomplished to provide value.
- Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)

- Process requirements describe activities performed by the developing organization. For instance, process requirements could specify .Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- **ECONOMIC FEASIBILITY**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

- **OPERATIONAL FEASIBILITY**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

- **TECHNICAL FEASIBILITY**

Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to .the users. The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

3.4.2. External Interface Requirements

User Interface

The user interface of this system is a user friendly python Graphical User Interface.

Hardware Interfaces

The interaction between the user and the console is achieved through python capabilities.

Software Interfaces

The required software is python.

SYSTEM REQUIREMENT:

HARDWARE REQUIREMENTS:

• Processor	-	Intel i3(min)
• Speed	-	1.1 GHz
• RAM	-	4GB(min)
• Hard Disk	-	500 GB
• Key Board	-	Standard Windows Keyboard
• Mouse	-	Two or Three Button Mouse
• Monitor	-	SVGA

SOFTWARE REQUIREMENTS:

• Operating System	-	Windows10(min)
• Programming Language	-	Python

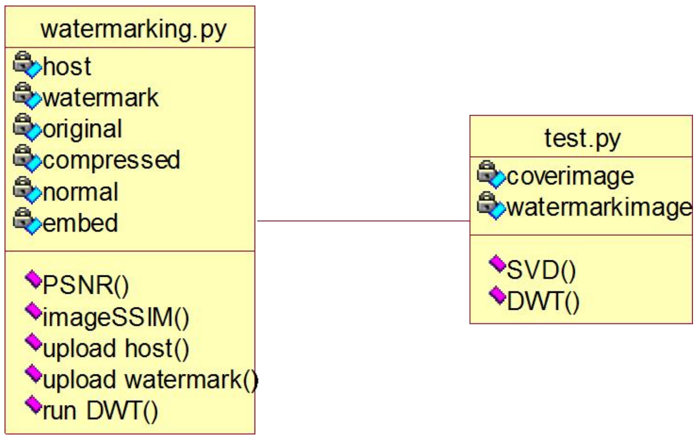
4. SYSTEM DESIGN

4. <u>SYSTEM DESIGN</u>

CLASS DIAGRAM:

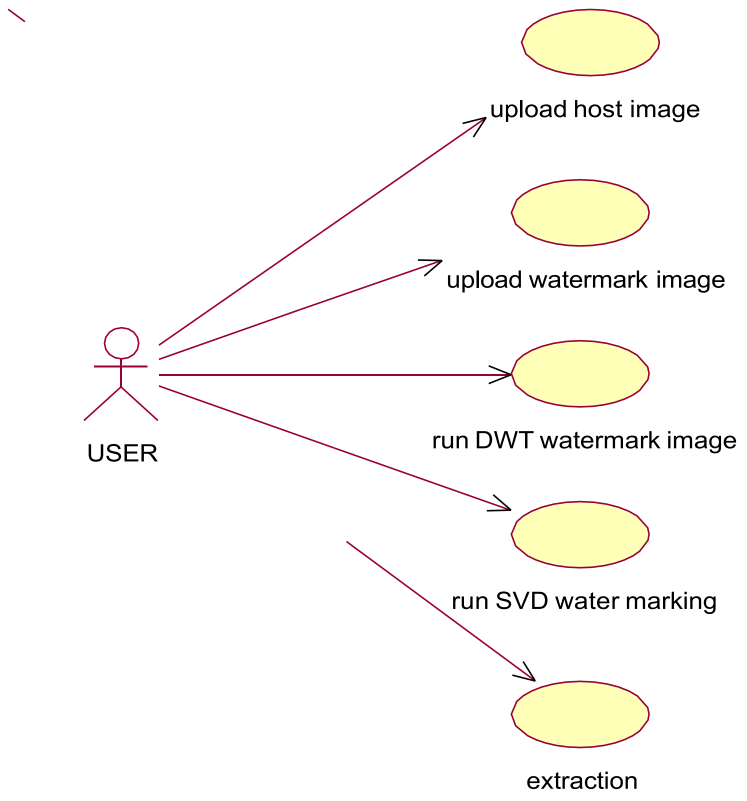
The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

- The upper part holds the name of the class
- The middle part contains the attributes of the class
- The bottom part gives the methods or operations the class can take or undertake



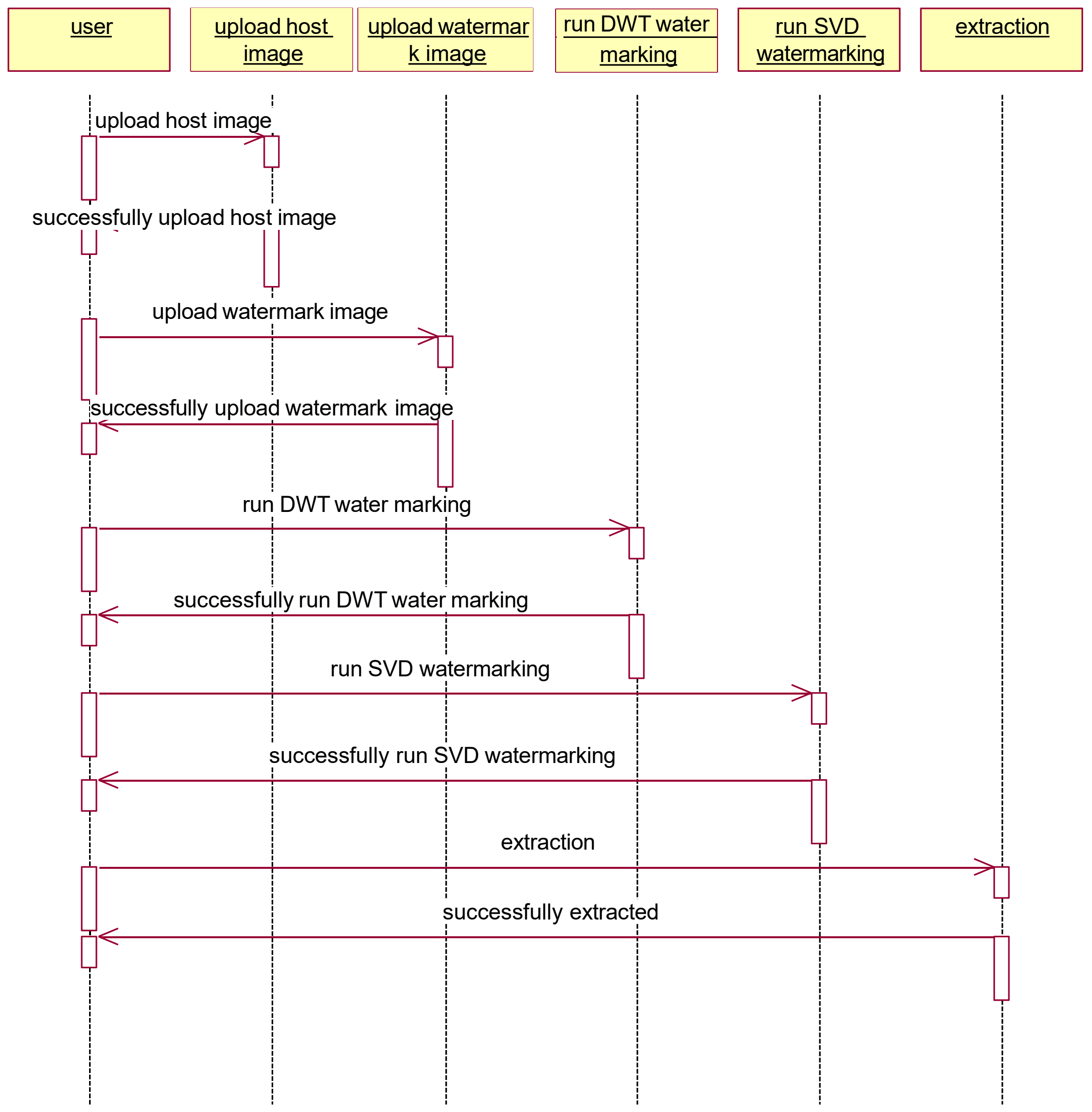
USECASE DIAGRAM:

A **use case diagram** at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.



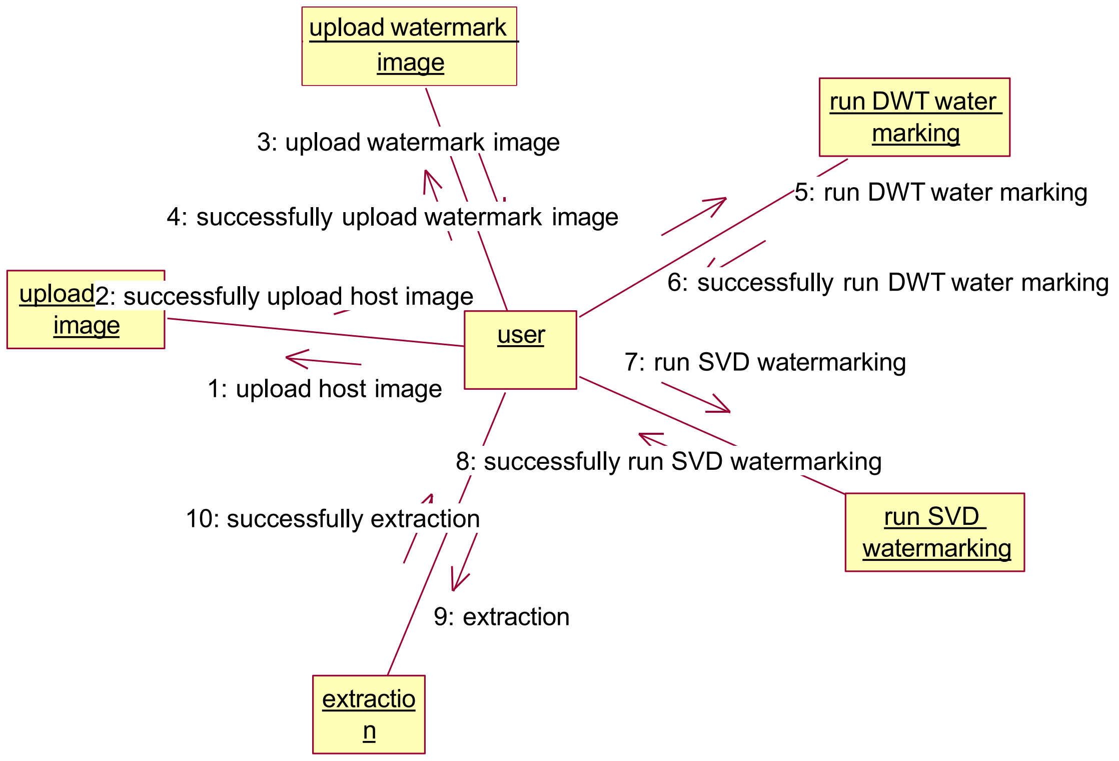
SEQUENCE DIAGRAM:

A **sequence diagram** is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams**, **event scenarios**, and timing diagrams.



COLLABORATION DIAGRAM:

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behaviour of a system.

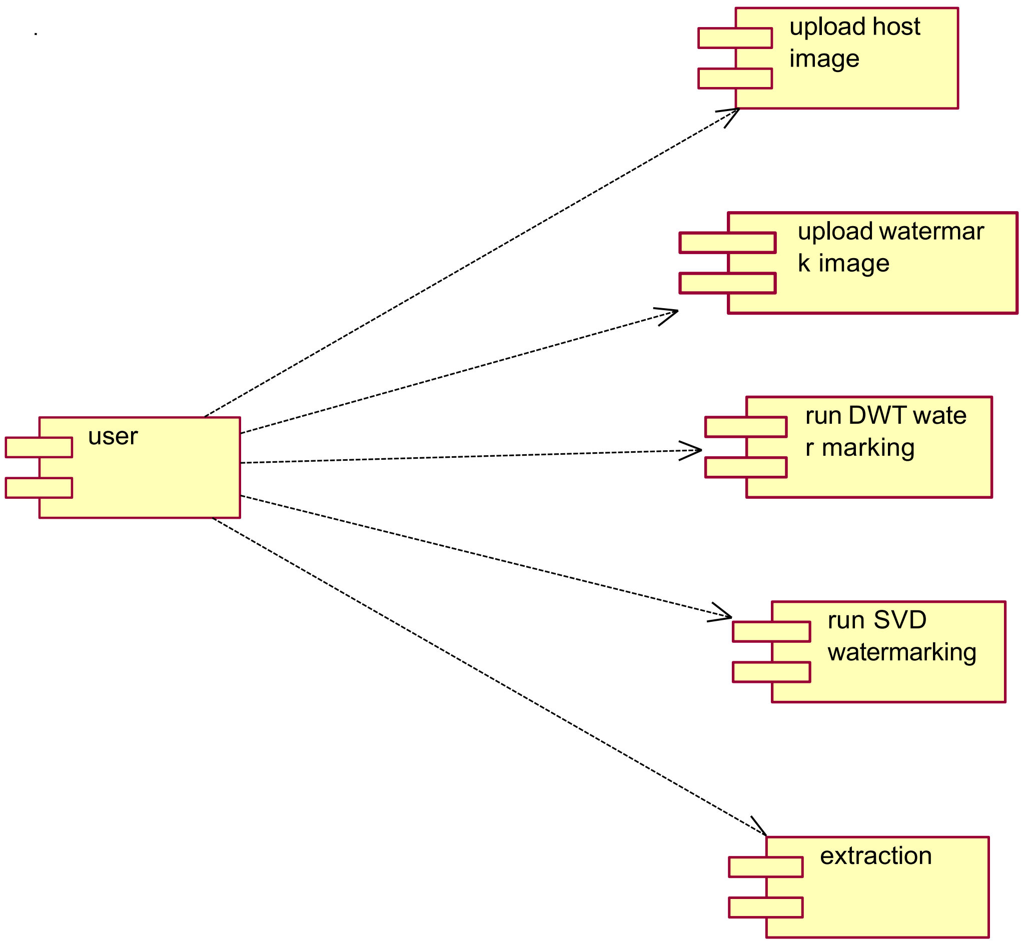


upload watermark image
run DWT water marking
upload image
user
run SVD watermarking
extraction
1: upload host image
2: successfully upload host image
3: upload watermark image
4: successfully upload watermark image
5: run DWT water marking
6: successfully run DWT water marking
7: run SVD watermarking
8: successfully run SVD watermarking
9: extraction
10: successfully extraction

COMPONENT DIAGRAM:

In the Unified Modelling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

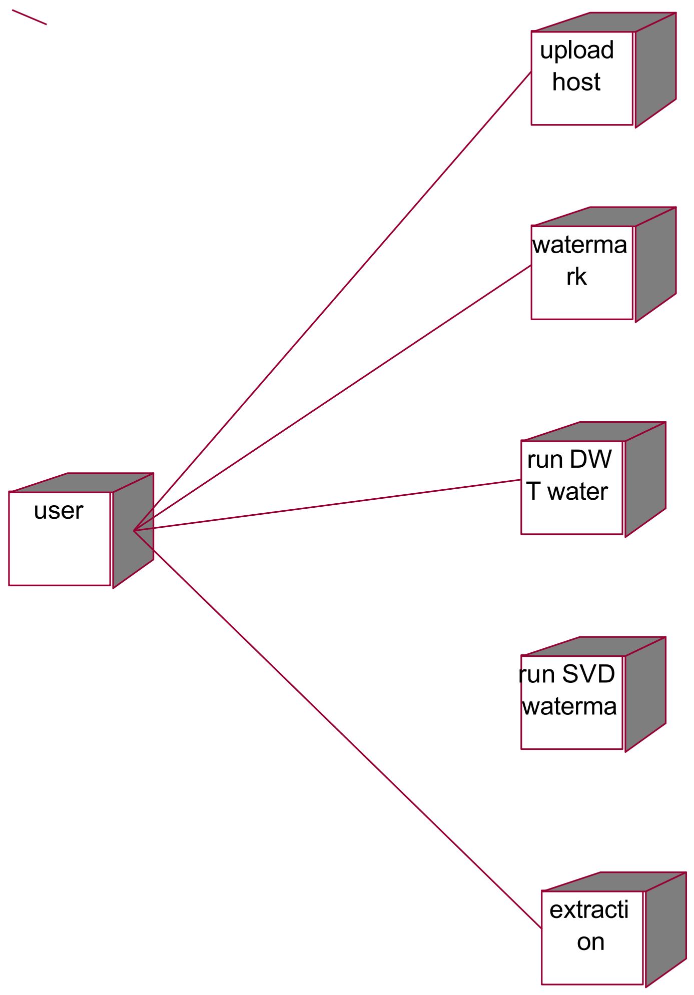
Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.



DEPLOYMENT DIAGRAM:

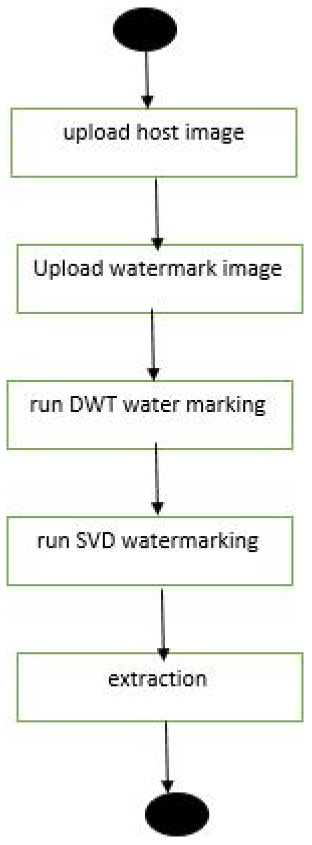
A **deployment diagram** in the Unified Modeling Language models the *physical* deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.



ACTIVITY DIAGRAM:

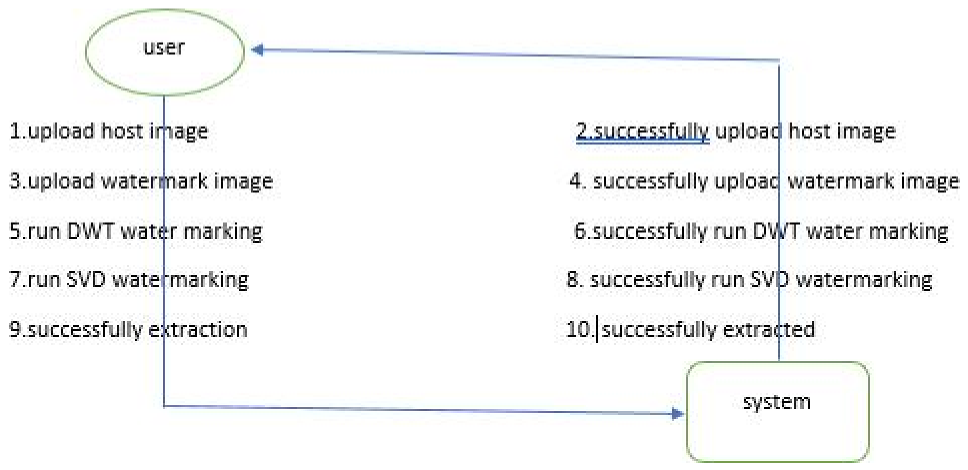
Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent.



Data flow :

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way.

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.


user
1.upload host image
3.upload watermark image
5.run DWT water marking
7.run SVD watermarking
9.successfully extraction
2.successfully upload host image
4. successfully upload watermark image
6.successfully run DWT water marking
8. successfully run SVD watermarking
10.successfully extracted
system

5. IMPLEMETATION

5.1 Python

Python is a general-purpose language. It has wide range of applications from Web development (like: Django and Bottle), scientific and mathematical computing (Orange, SymPy, NumPy) to desktop graphical user Interfaces (Pygame, Panda3D). The syntax of the language is clean and length of the code is relatively short. It's fun to work in Python because it allows you to think about the problem rather than focusing on the syntax.

History of Python:

Python is a fairly old language created by Guido Van Rossum. The design began in the late 1980s and was first released in February 1991.

Why Python was created?

In late 1980s, Guido Van Rossum was working on the Amoeba distributed operating system group. He wanted to use an interpreted language like ABC (ABC has simple easy-to-understand syntax) that could access the Amoeba system calls. So, he decided to create a language that was extensible. This led to design of a new language which was later named Python.

Why the name Python?

No. It wasn't named after a dangerous snake. Rossum was fan of a comedy series from late seventies. The name "Python" was adopted from the same series "Monty Python's Flying Circus".

Features of Python:

A simple language which is easier to learn

Python has a very simple and elegant syntax. It's much easier to read and write Python programs compared to other languages like: C++, Java, C#. Python makes programming fun and allows you to focus on the solution rather than syntax.

If you are a newbie, it's a great choice to start your journey with Python.

Free and open-source

You can freely use and distribute Python, even for commercial use. Not only can you use and distribute software's written in it, you can even make changes to the Python's source code.

Python has a large community constantly improving it in each iteration.

Portability

You can move Python programs from one platform to another, and run it without any changes.

It runs seamlessly on almost all platforms including Windows, Mac OS X and Linux.

Extensible and Embeddable

Suppose an application requires high performance. You can easily combine pieces of C/C++ or other languages with Python code.

This will give your application high performance as well as scripting capabilities which other languages may not provide out of the box.

A high-level, interpreted language

Unlike C/C++, you don't have to worry about daunting tasks like memory management, garbage collection and so on.

Likewise, when you run Python code, it automatically converts your code to the language your computer understands. You don't need to worry about any lower-level operations.

Large standard libraries to solve common tasks

Python has a number of standard libraries which makes life of a programmer much easier since you don't have to write all the code yourself. For example: Need to connect MySQL database on a Web server? You can use MySQLdb library using import MySQLdb .

Standard libraries in Python are well tested and used by hundreds of people. So you can be sure that it won't break your application.

Object-oriented

Everything in Python is an object. Object oriented programming (OOP) helps you solve a complex problem intuitively.

With OOP, you are able to divide these complex problems into smaller sets by creating objects.

Applications of Python:

1. Simple Elegant Syntax

Programming in Python is fun. It's easier to understand and write Python code. Why? The syntax feels natural. Take this source code for an example:

```
a = 2

b = 3

sum = a + b

print(sum)
```

2. Not overly strict

You don't need to define the type of a variable in Python. Also, it's not necessary to add semicolon at the end of the statement.

Python enforces you to follow good practices (like proper indentation). These small things can make learning much easier for beginners.

3. Expressiveness of the language

Python allows you to write programs having greater functionality with fewer lines of code. Here's a link to the source code of Tic-tac-toe game with a graphical interface and a smart computer opponent in less than 500 lines of code. This is just an example. You will be amazed how much you can do with Python once you learn the basics.

4. Great Community and Support

Python has a large supporting community. There are numerous active forums online which can be handy if you are stuck.

5.2 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("Watermarking Images")
main.geometry("1200x1200")
```

```
global host, watermark

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

def imageSSIM(normal, embed):
    #original = cv2.cvtColor(original, cv2.COLOR_BGR2GRAY)
    #super_image = cv2.cvtColor(super_image, cv2.COLOR_BGR2GRAY)
    ssim_value = ssim(normal, embed, data_range = embed.max() - embed.min())
    return ssim_value
```

```
def uploadHost():
    global host
    text.delete('1.0', END)
    host = filedialog.askopenfilename(initialdir = "hostImages")
    pathlabel.config(text=host+" host image loaded")

def uploadWatermark():
    global watermark
    watermark = filedialog.askopenfilename(initialdir = "watermarkImages")
    pathlabel.config(text=watermark+" watermark image loaded")


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

    #Embedding
```

```
 coeffW = (0.4*cA + 0.1*watermarkImage, (cH, cV, cD))
  watermarkedImage = pywt.idwt2(coeffW, 'haar')
  psnr,mse = PSNR(coverImage,watermarkedImage)
  ssim = imageSSIM(coverImage,watermarkedImage)
  text.insert(END,"DWT PSNR : "+str(psnr)+"\n")
  text.insert(END,"DWT MSE  : "+str(mse)+"\n")
  text.insert(END,"DWT SSIM : "+str(ssim)+"\n\n")
  text.update_idletasks()
  name = os.path.basename(host)
  cv2.imwrite("OutputImages/"+name,watermarkedImage*255)
  np.save("model/"+name,watermarkedImage)
```

```
    np.save("model/CA_"+name,cA)
    cv2.imshow('Watermarked Image', cv2.resize(watermarkedImage,(400,400)))
    cv2.waitKey(0)

def runExtraction():
    wm = filedialog.askopenfilename(initialdir = "OutputImages")
    wm = os.path.basename(wm)
    img = np.load("model/"+wm+".npy")
    cA = np.load("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)
```

```
def runSVD():
    coverImage = cv2.imread(host,0)
    watermarkImage = cv2.imread(watermark,0)
    cv2.imshow('Cover Image',cv2.resize(coverImage,(400,400)))
    [m,n]=np.shape(coverImage)
    coverImage=np.double(coverImage)
    cv2.imshow('Watermark Image',cv2.resize(watermarkImage,(400,400)))
    watermarkImage = np.double(watermarkImage)

    #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(0,x):
  for j in range(0,y):
      Wcvr[i,j]=(Wcvr[i,j]+0.01*watermarkImage[i,j])/255

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

#Watermarked Image
S=np.zeros((512,512),np.uint8)
print(str(S.shape)+" "+str(m)+" "+str(n))
S[:m,:n]=np.diag(w)
S=np.double(S)
wimg=np.matmul(ucvr,np.matmul(S,vtcvr))
```

```
wimg=np.double(wimg)
wimg*=255
watermarkedImage = np.zeros(wimg.shape,np.double)
normalized=cv2.normalize(wimg,watermarkedImage,1.0,0.0,cv2.NORM_MINMAX)
psnr,mse = PSNR(coverImage,watermarkedImage)
ssim = imageSSIM(coverImage,watermarkedImage)
text.insert(END,"DWT PSNR : "+str(psnr)+"\n")
text.insert(END,"DWT MSE  : "+str(mse)+"\n")
text.insert(END,"DWT SSIM : "+str(ssim)+"\n\n")
text.update_idletasks()
```

```
    watermarkedImage = cv2.resize(watermarkedImage,(400,400))
    cv2.imshow('Watermarked Image',watermarkedImage)
    cv2.waitKey(0)



font = ('times', 20, 'bold')
title = Label(main, text='Watermarking Images')
title.config(bg='brown', fg='white')
title.config(font=font)
title.config(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 DWT Watermarking", command=runDWT)
dwtButton.place(x=50,y=200)
dwtButton.config(font=font1)
```

```
svdButton = Button(main, text="Run SVD Watermarking", 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()
```

6. TESTING:

Implementation and Testing:

Implementation is one of the most important tasks in project is the phase in which one has to be cautions because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user.

Implementation

The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modifies as a result of a programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

Testing

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. Actually testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period.

System Testing

Testing has become an integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to use the software must be

tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

Module Testing

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

Integration Testing

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system.

Acceptance Testing

When that user fined no major problems with its accuracy, the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

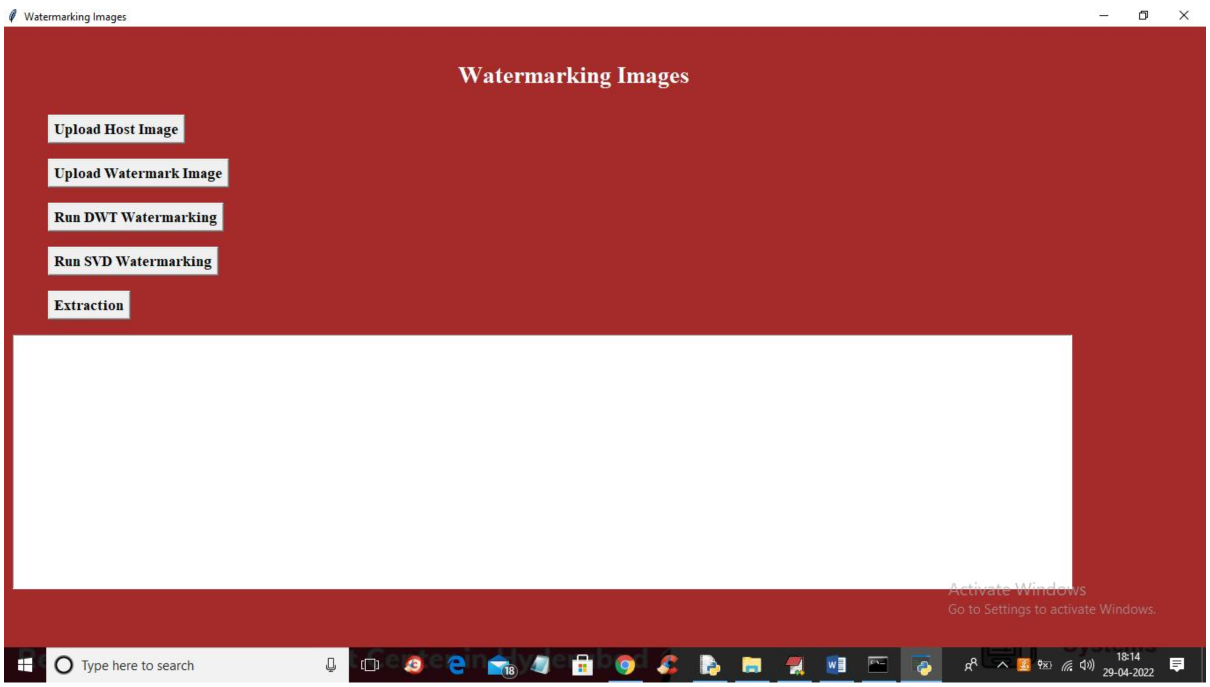
Test Case Id	**Test Case Name**	**Test Case Desc.**	**Test Steps**			**Test Case Status**	**Test Priority**
			Step	**Expected**	**Actual**		
01	upload host image	Verify host image Uploadedor not	If User host image may not upload	we cannot do any further operations	we can do further operations	High	High

02	watermark image	Verify watermark image or not	If watermark image may not be Done	we cannot do any further operations	we can do further operations	High	High
03	run DWT water marking	Verify DWT water marking run or not	If DWT water marking may not be done	we cannot do any further operations	we can do further operations	High	High
04	run SVD watermarking	Verify Run SVD watermarking or not	If SVD watermarking may not Run	We cannot run operation	We can Run the Operation	High	High
05	Extraction	Verify Data Features Extraction or not	If Features Extraction may not be Done	we cannot do any further operations	we can do further operations	High	High

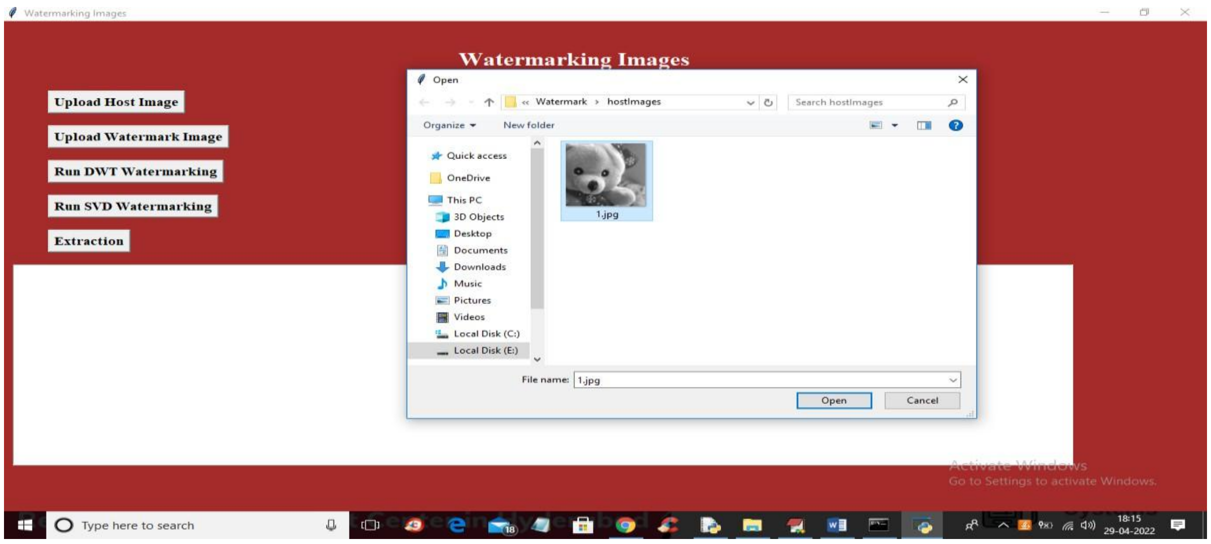
7. SCREENSHOTS:

In this project we have implemented SVD and DWT technique to embed watermark images inside cover image and then calculate SSIM, PSNR and MSE between cover image and watermark embedded image.

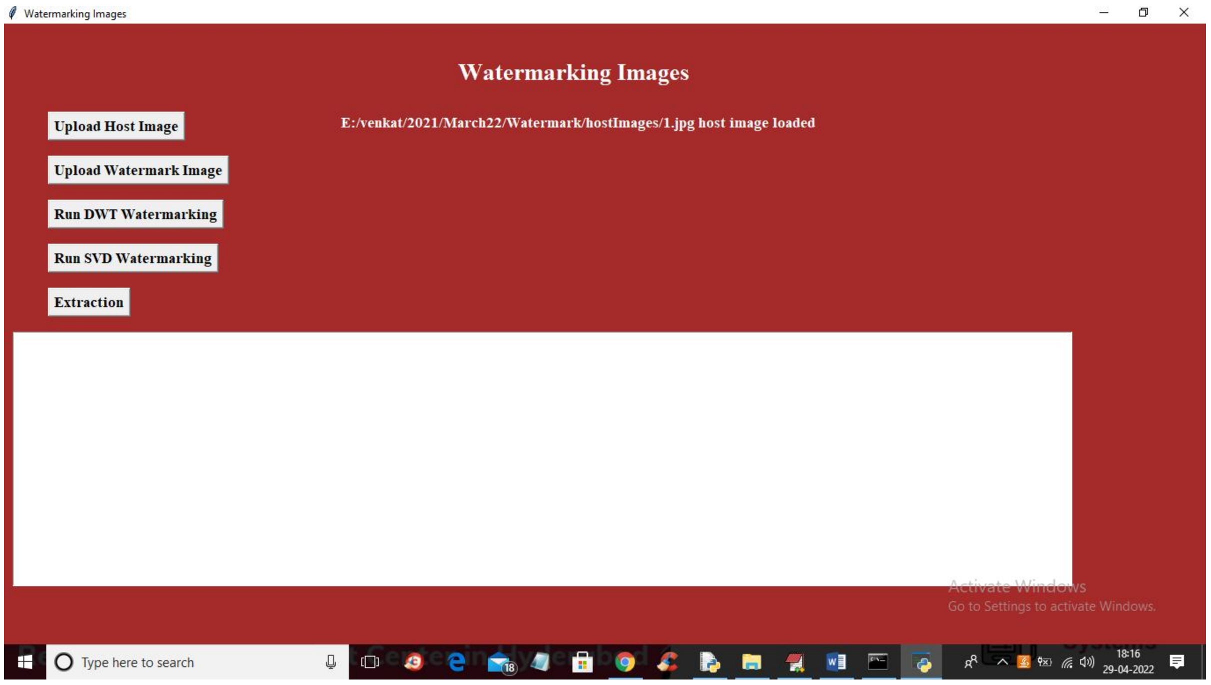
To run project double click on ‘run.bat’ file to get below screen



In above screen click on 'Upload Host Image' button to upload host or cover image like below screen

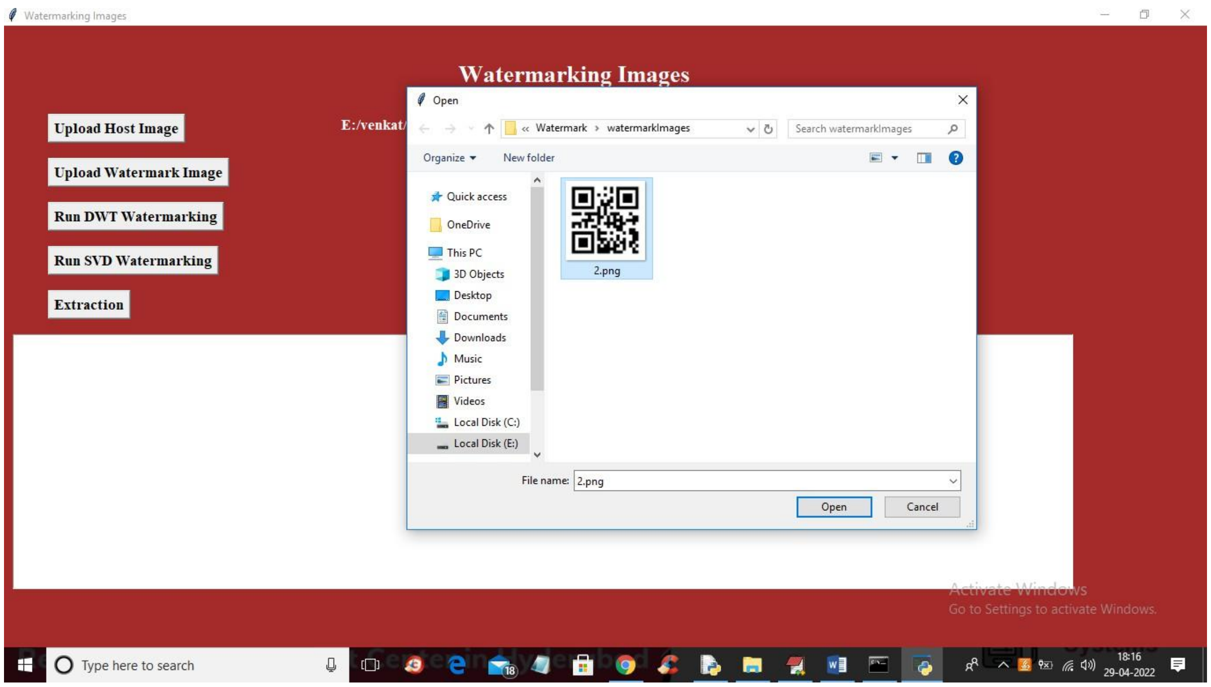


In above screen selecting and uploading 1.jpg as cover image and you can image from any folder or put your desired images in this folder and upload and now click 'Open' button to get below screen

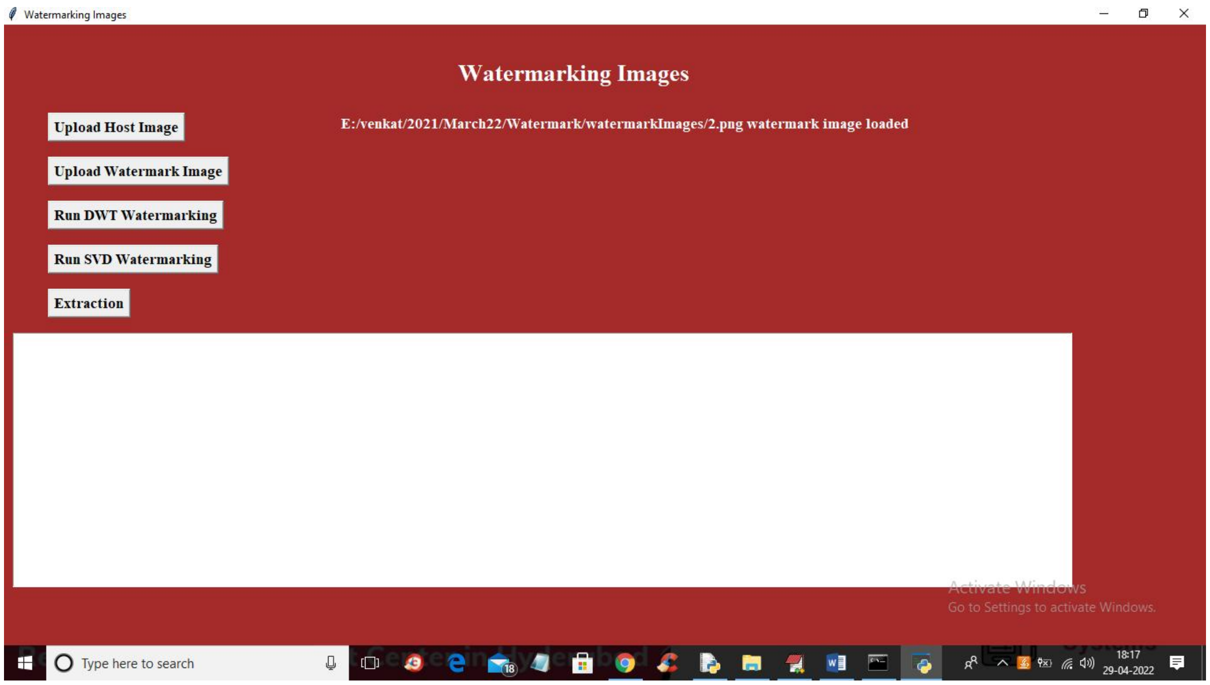


E:/venkat/2021/March22/Watermark/hostImages/1.jpg host image loaded

In above screen cover image is loaded and now click on 'Upload Watermark Image' button to upload watermark image

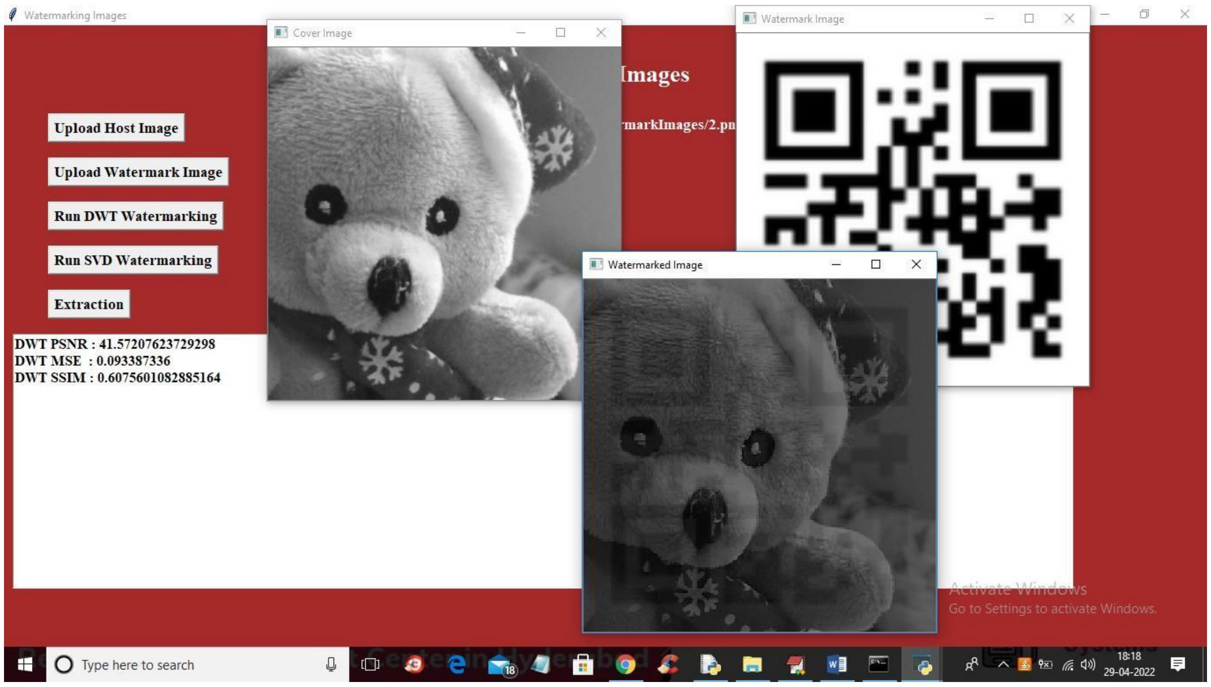


In above screen selecting and uploading watermark image and then click on ‘Open’ button to load image and get below output



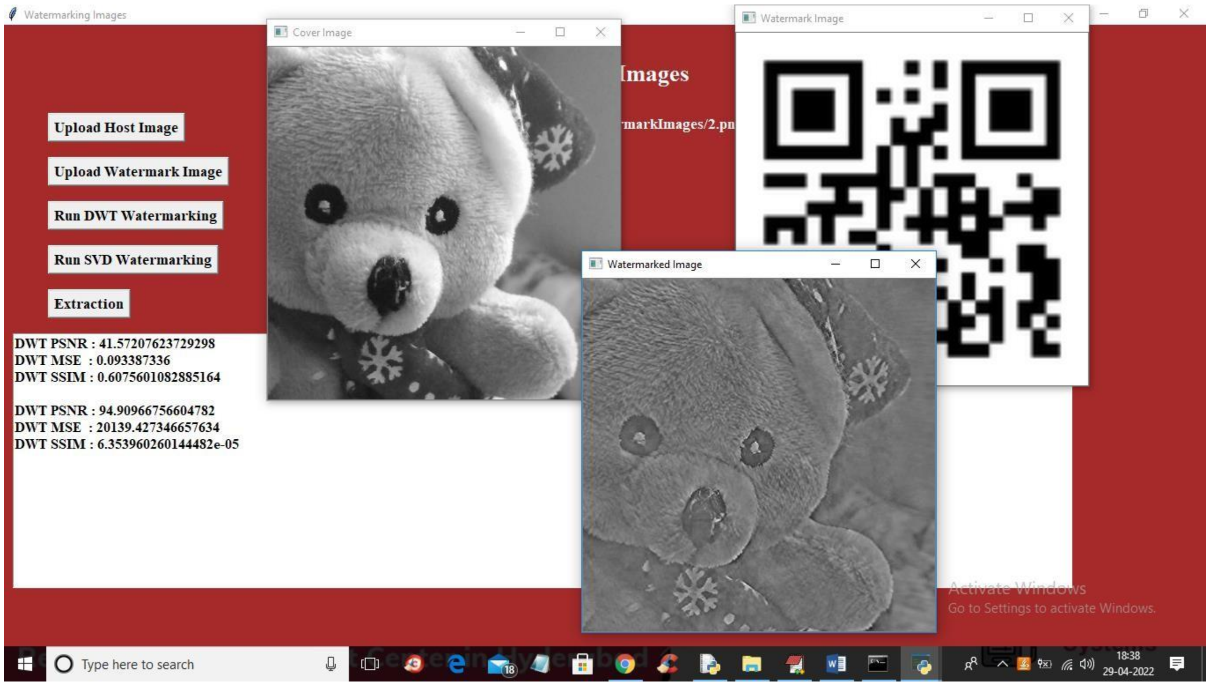
E:/venkat/2021/March22/Watermark/watermarkImages/2.png watermark image loaded

In above screen in white colour text we can see watermark image is loaded and now click on 'Run DWT Watermarking' button to embed image and get below output



```
DWT PSNR : 41.57207623729298
DWT MSE  : 0.093387336
DWT SSIM : 0.6075601082885164
```

In above screen first image is the cover image and second is the watermarking image and 3r is the embedded watermarked image and we can see DWT PSNR and other values in the text area for DWT. Now click on ‘Run SVD Watermarking’ button to embed watermarking using SVD

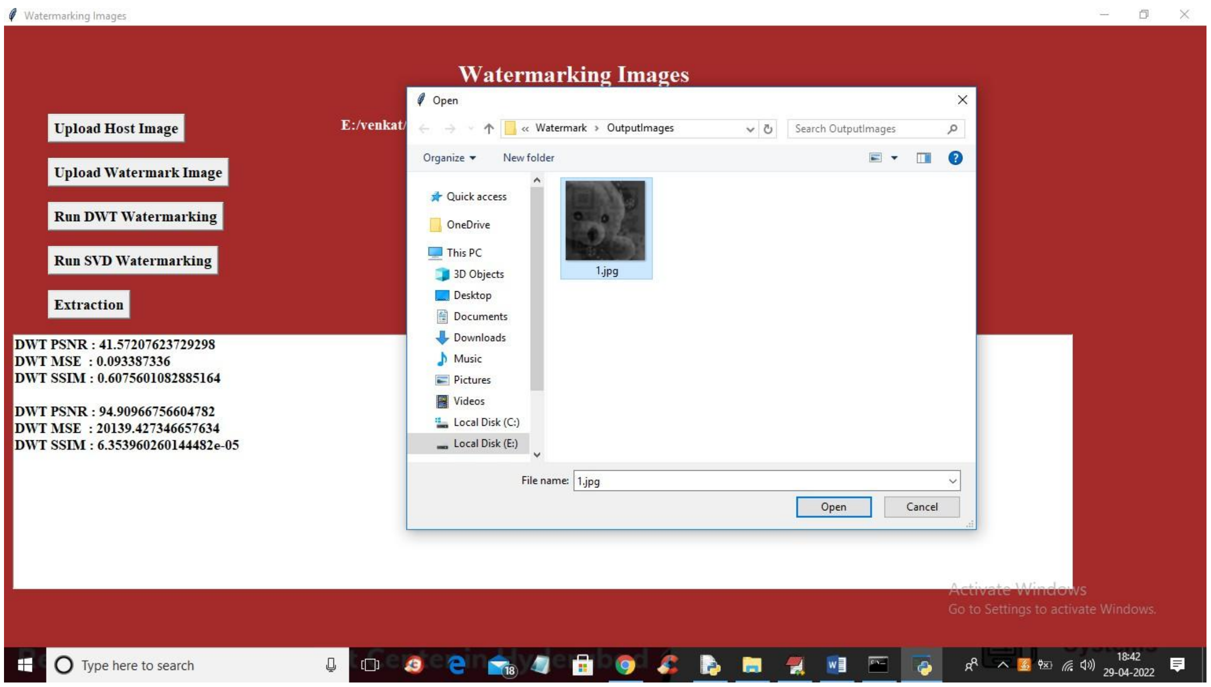


```
DWT PSNR : 41.57207623729298
DWT MSE  : 0.093387336
DWT SSIM : 0.6075601082885164

DWT PSNR : 94.90966756604782
DWT MSE  : 20139.427346657634
DWT SSIM : 6.353960260144482e-05
```

In above screen first image is the original image and second is the water mark image and 3rd is the embedded water marked image using SVD algorithm and we can see PSNR, MSE and SSIM for both SVD and DWT. PSNR must be closer to 100% to consider as high quality image and MSE must be closer to 0 and SSIM must be closer

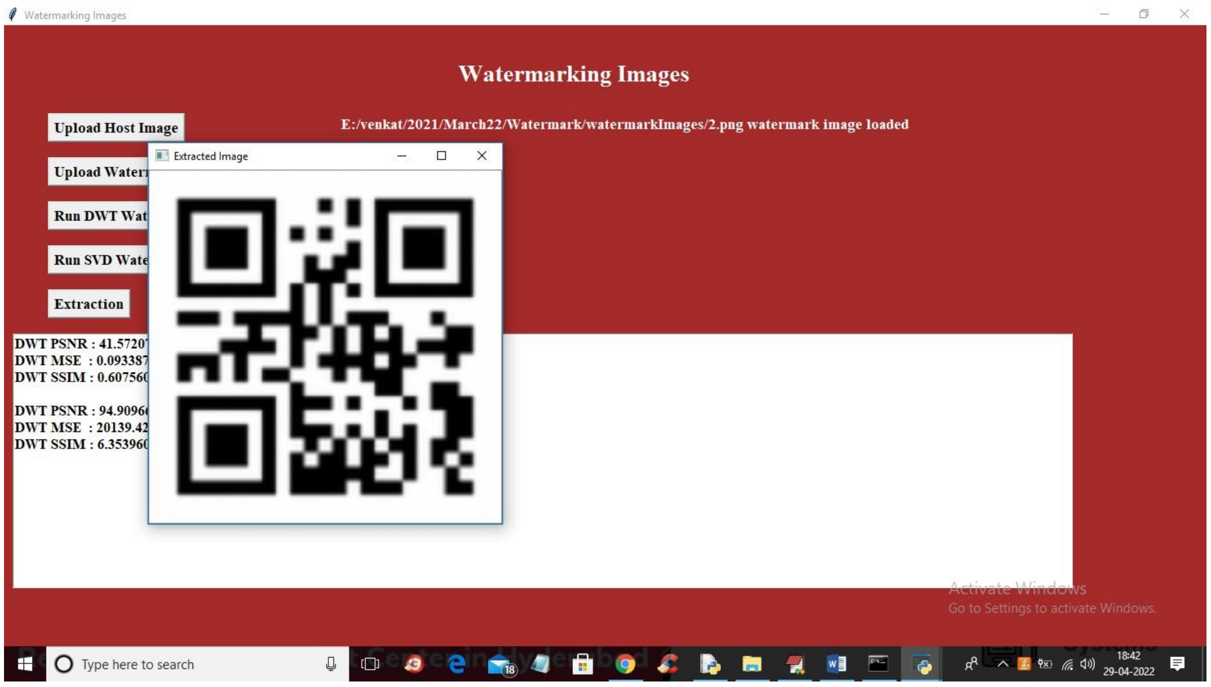
to 100. Now click on ‘Extraction’ button to upload Watermarked image and then extract embedded image from it



```
DWT PSNR : 41.57207623729298
DWT MSE  : 0.093387336
DWT SSIM : 0.6075601082885164

DWT PSNR : 94.90966756604782
DWT MSE  : 20139.427346657634
DWT SSIM : 6.353960260144482e-05
```

In above screen selecting and uploading embedded watermark image and then click on ‘Open’ button to get below output



E:/venkat/2021/March22/Watermark/watermarkImages/2.png watermark image loaded

In above screen we can see the extracted image and similarly you can upload any image and get output

8. CONCLUSION:

The proliferation of network multimedia systems dictates the need for copyright protection of digital property. This paper presents a visually undetectable, robust watermarking scheme. Our techniques can detect the change of a single pixel and can locate where the changes occur. The algorithms work for color images and can accommodate PEG compression. Future research includes watermarking MPEG video sequences.

9. REFERENCES:

[l] S. Walton, "Information authentication for a slippery new age," Dr. Dobbs Journal, vol. 20, no. 4, pp. 18-26, April, 1995.

[2] R. G. van Schyndel, A. 2. Tirkel, N. R. A. Mee, C. F. Osborne, "A digital watermark," Proceedings of the International Conference on Image Processing, November, 1994, Austin, Texas, vol. 2, pp. 86-90.

[3] J.-F. Delaigle, C. De Vleeschouwer, B. Macq, "Digital watermarking," accepted for publication, Journal of Electronic Imaging

[4] F. M. Boland, J. J. K. 0 Ruanaidh and C. Dautzenbcrg, "Watermarking digital images for copyright protection," Proceedings of the International Conference on Image Processing and its Applications, July 1995, Edinburgh, Scotland, pp. 321-326.

[5] Fred Mintzer, Albert Cazes, Francis Giordano, Jack Lee, Karen Magerlein and Fabio Schiattarella, "Capturing and preparing images of Vatican library manuscripts for access via internet," Proceedings of ISdTS 48th Annual Conference, May, 1995, Washington, DC, pp. 74 - 77.

[6] J. Proakis, Digital Communications, McGraw-Hill, 1983.