Industry Oriented Mini Project Report

# SECURING DATA IN IMAGE AND AUDIO FILES USING STEGANOGRAPHY TECHNIQUE

Submitted in partial fulfillment of the requirements for the award of the Degree of **Bachelor of Technology**

in

COMPUTER SCIENCE AND ENGINEERING

By

| | |
|---|---|
| **KONDURU CHATHURYA** | **16241A05Q9** |
| **PERUMALLA RAMYA** | **16241A05R8** |
| **PERUMANDLA JAYASREE** | **16241A05R9** |
| **VASIREDDY SANJANA** | **16241A05T5** |

Under the Esteemed guidance of

**DR. PADMALAYA NAYAK**

**Professor**

**Department of Computer Science and Engineering**

***GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING AND TECHNOLOGY***

**(Autonomous)**

**Bachupally, Kukatpally, Hyderabad- 500090**

**2018-2019**

***GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING AND TECHNOLOGY***

**(Autonomous)**

**Department of Computer Science and Engineering**

## CERTIFICATE

This is to certify that the Industrial mini project entitled **"SECURING DATA IN IMAGE AND AUDIO FILES USING STEGANOGRAPHY TECHNIQUE"** is submitted by **K.CHATHURYA (16241A05Q7), P.RAMYA(16241A05R8), P.JAYASREE (16241A05R9), V.SANJANA (16241A05T5)** in partial fulfillment of the requirement for the award of the degree in **BACHELOR OF TECHNOLOGY** in Computer Science and Engineering during the academic year 2018-2019.

| | |
|---|---|
| INTERNAL GUIDE | HEAD OF THE DEPARTMENT |
| **PROF.PADMALAYA NAYAK** | **DR.K.MADHAVI** |

EXTERNAL EXAMINER

# DECLARATION

I hereby declare that the industrial mini project entitled **"SECURING DATA IN IMAGE AND AUDIO FILES USING STEGANOGRAPHY TECHNIQUE"** is the work done during the period from 10th December 2018 to 4th April 2019 and is submitted in the partial fulfillment of the requirements for the award of degree of Bachelor of Technology in Computer Science and Engineering from Gokaraju Rangaraju Institute of Engineering and Technology (Autonomous under Jawaharlal Nehru Technology University, Hyderabad). The results embodied in this project have not been submitted to any other University or Institution for the award of any degree or diploma.

**K.CHATHURYA - 16241A05Q7**

**P.RAMYA - 16241A05Q8**

**P.JAYASREE- 16241A05S7**

**V.SANJANA - 17245A0550**

# ACKNOWLEDGEMENT

There are many people who helped us directly and indirectly to complete our project successfully. We would like to take this opportunity to thank one and all.

First of all we would like to express our deep gratitude towards our internal guide **DR. PADMALAYA NAYAK, Professor,** Department of CSE for her support in the completion of my dissertation. We wish to express our sincere thanks to **Dr. K.Madhavi, HOD, Department of CSE** and our mini project coordinator **SRIKANTH** for her external support. We also thank our principal **Dr. J.Praveen** for providing the facilities to complete the dissertation.

We would like to thank all our faculty and friends for their help and constructive criticism during the project period. Finally, we are very much indebted to our parents for their moral support and encouragement to achieve our goals.

**K.CHATHURYA - 16241A05Q7**

**P.RAMYA - 16241A05R8**

**P.JAYASREE - 16241A05R9**

**V.SANJANA - 16241A05T5**

# Securing Data in Image & Audio Files using Stegenography Technique

## Abstract

Steganography is the art of hiding information within other information in such a way that it is hard to identify the existence of any hidden information. There are many different carries. Of which, most popular ones are digital images. Due to recent developments in steganalysis, providing security to personal contents, messages, digital images using steganography has become difficult. By using steganalysis, one can easily reveal existence of hidden information in carrier files. This project introduces a novel steganographic approach for covert communications between two private parties. The approach introduced in this project makes use of both steganographic as well as cryptographic techniques. The process involves converting a secret image into a text document, then encrypting the generated text into a cipher text using a key (password) based encryption algorithm, finally embedding the cipher text on to a cover image. This embedding process is carried out using a threshold based scheme that inserts secret message bits in to the cover image only in selected pixels. The security to maintain secrecy of message is achieved by making it infeasible for a third person to detect and retrieve the hidden message.

**Keywords**: Steganography, Steganalysis, Cryptography, Cipher text, Encryption

## CONTENTS Page No

# 1. INTRODUCTION

## 1.1 INTRODUCTION:

In open networks, to securely transmit data Encryption and Decryption is used. To protect confidential image data from unauthorized access as each type of data has its own features, different techniques should be used. The principle goal of designing any encryption and decryption algorithm is to hide the original message and send the non-readable text message to the receiver so that secret message communication can take place over the web. Algorithm strength depends on the difficulty of cracking the original message. Data encryption is widely used to ensure security however; most of the available encryption algorithm is used for text data. Encryption is the process to convert original image data in to some other anonymous structure using a key which is not identified by anyone. Decryption defines the recovery of original data from the encrypted thing.

Illegal copying, modifying, tampering and copyright protection have become very important issues with the rapid use of internet [7]. Hence, there is a strong need of developing the techniques to face all these problems. Digital watermarking [1] emerged as a solution for protecting the multimedia data. Digital Watermarking is the process of hiding or embedding an imperceptible signal (data) into the given signal (data). This imperceptible signal (data) is called watermark or metadata and the given signal (data) is called cover work. The watermark should be embedded into the cover work, so that it should be robust enough to survive not only the most common signal distortions, but also distortions caused by malicious attacks. This cover work can be an image, audio or a video file. A watermarking algorithm consists of two algorithms, an embedding and an extraction (or detection) algorithm. The idea of watermarking first appeared hundreds of years ago [2]. Watermarking technology was used to mark information authenticity by many different means. Watermarking technology has been used in computer as well. Most of the work on computer watermarking technology was for embedding a watermark into images, audio, and video files. Media watermarking research is a very active area and digital image watermarking became an interesting protection measure and got the attention of many researchers since the early 1990s [3].

# 2. SYSTEM ANALYSIS

The Systems Development Life Cycle (SDLC), or Software Development Life Cycle in systems engineering, information systems and software engineering, is the process of creating or altering systems, and the models and methodologies that people use to develop these systems. In software engineering the SDLC concept underpins many kinds of software development methodologies.

## 2.1 EXISTING SYSTEM:

Existing system is a manual & time taking process.

## 2.3 PROPOSED SYSTEM:

In This Project we have to hide the data in Image and also data hiding in Audio Files by using LSB encode, decode methods.

# 3. IMPLEMENTATION

## 3.1 Introduction of Technologies Used

Initially Java language was called as “oak” but it was renamed as “java” in 1995.The primary motivation of this language was the need for a platform-independent i.e. architecture neutral language that could be used to create software to be embedded in various consumer electronic devices.

**Applications and applets**

An application is a program that runs on our Computer under the operating system of that computer. It is more or less like one creating using C or C++ .Java’s ability to create Applets makes it important. An Applet I san application, designed to be transmitted over the Internet and executed by a Java-compatible web browser. An applet I actually a tiny Java program, dynamically downloaded across the network, just like an image. But the difference is, it is an intelligent program, not just a media file. It can be react to the user input and dynamically change.

**Java Architecture**

Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

## Compilation of code

When we compile the code, the Java compiler creates machine code called byte code for a hypothetical machine called Java Virtual Machine (JVM). Compiling and interpreting java source code.

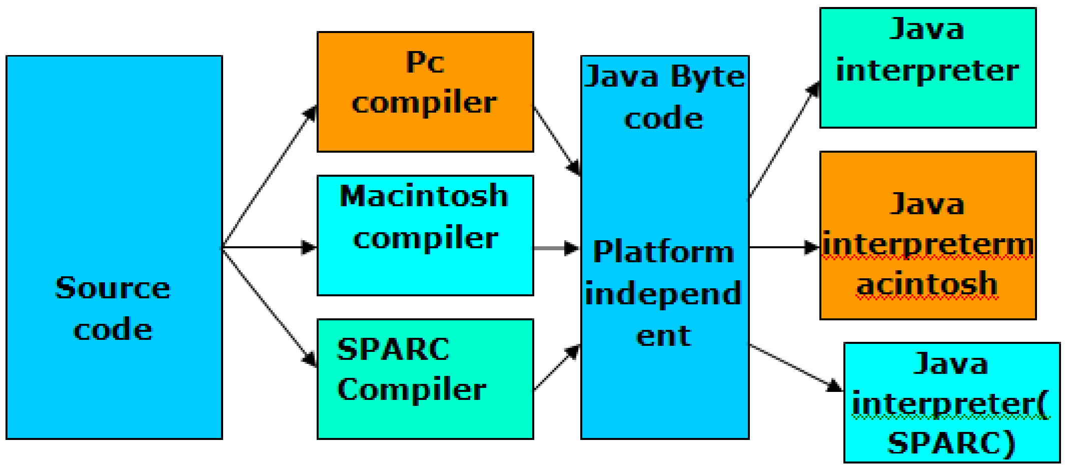


**Fig 4.1: Structure of compilation**

During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine. In reality this could be an Intel Pentium windows 95 or sun SPARCstation running Solaris or Apple Macintosh running system and all could receive code from any computer through internet and run the Applets.

## Simple

Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ Programmer, learning Java will oriented features of C++. Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task.

**Object oriented**

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank state. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

**Robust**

The multi-platform environment of the web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs was given a high priority in the design of Java. Java is strictly typed language; it checks the code at compile time and runtime.
Java virtually eliminates the problems of memory management and deal location, which is completely automatic. In a well-written Java program, all run-time errors can and should be managed by your program.

# 4. SOFTWARE REQUIREMENT SPECIFICATION

## 4.1 Requirements Specification:

Requirement Specification provides a high secure storage to the web server efficiently. Software requirements deal with software and hardware resources that need to be installed on a serve which provides optimal functioning for the application. These software and hardware requirements need to be installed before the packages are installed. These are the most common set of requirements defined by any operation system. These software and hardware requirements provide a compatible support to the operation system in developing an application.

### 4.1.1 HARDWARE REQUIREMENTS:

The hardware requirement specifies each interface of the software elements and the hardware elements of the system. These hardware requirements include configuration characteristics.

| | |
|---|---|
| ➢ System | : Pentium IV 2.4 GHz. |
| ➢ Hard Disk | : 100 GB. |
| ➢ Monitor | : 15 VGA Color. |
| ➢ Mouse | : Logitech. |
| ➢ RAM | : 1 GB. |

### 4.1.2 SOFTWARE REQUIREMENTS:

The software requirements specify the use of all required software products like data management system. The required software product specifies the numbers and version. Each interface specifies the purpose of the interfacing software as related to this software product.

| | | |
|---|---|---|
| ➢ Operating system | : | Windows XP/7/10 |
| ➢ Coding Language | : | Html, JavaScript, Java/J2EE (Jsp Servlet) |
| ➢ Development Kit | : | JDK 1.7 |
| ➢ Database | : | MySQL |
| ➢ IDE | : | Netbeans |
| ➢ Server | : | Tomcat 7.0 |

## 4.2 FUNCTIONAL REQUIREMENTS:

The functional requirement refers to the system needs in an exceedingly computer code engineering method.

The key goal of determinant “functional requirements” in an exceedingly product style and implementation is to capture the desired behavior of a software package in terms of practicality and also the technology implementation of the business processes.

## 4.3 NON FUNCTIONAL REQUIREMENTS

All the other requirements which do not form a part of the above specification are categorized as Non-Functional needs. A system perhaps needed to gift the user with a show of the quantity of records during info. If the quantity must be updated in real time, the system architects should make sure that the system is capable of change the displayed record count at intervals associate tolerably short interval of the quantity of records dynamic. Comfortable network information measure may additionally be a

non-functional requirement of a system.

The following are the features:

- Accessibility
- Availability
- Backup
- Certification
- Compliance
- Configuration Management
- Documentation
- Disaster Recovery
- Efficiency(resource consumption for given load)
- Interoperability

## 4.4 PERFORMANCE REQUIREMENTS

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely with the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

- The system should be able to interface with the existing system
- The system should be accurate
- The system should be better than the existing system

The existing system is completely dependent on the user to perform all the duties.

## 4.5 Feasibility Study:

Preliminary investigation examines 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 systems are feasible if they are given unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Operation Feasibility

Economical Feasibility

### 4.5.1 Technical Feasibility

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Do the proposed equipments have the technical capacity to hold the data required to use the new system?
- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- Can the system be upgraded if developed?

Are there technical guarantees of accuracy, reliability, ease of access and data security?

### 4.5.2 Operational Feasibility

**User-friendly**

Customer will use the forms for their various transactions i.e. for adding new routes, viewing the routes details. Also the Customer wants the reports to view the various transactions based on the constraints. These forms and reports are generated as user-friendly to the Client.

**Reliability**

The package wills pick-up current transactions on line. Regarding the old transactions, User will enter them in to the system.

**Security**

The web server and database server should be protected from hacking, virus etc

**Portability**

The application will be developed using standard open source software (Except Oracle) like Java, tomcat web server, Internet Explorer Browser etc these software will work both on Windows and Linux o/s. Hence portability problems will not arise.

**Availability**

This software will be available always.

**Maintainability**

The system uses the 2-tier architecture. The 1st tier is the GUI, which is said to be front-end and the 2nd tier is the database, which uses My-Sql, which is the back-end.

The front-end can be run on different systems (clients). The database will be running at the server. Users access these forms by using the user-ids and the passwords.

### 4.5.3 Economic Feasibility

The computerized system takes care of the present existing system's data flow and procedures completely and should generate all the reports of the manual system besides a host of other management reports.

It should be built as a web based application with separate web server and database server. This is required as the activities are spread throughout the organization customer wants a centralized database. Further some of the linked transactions take place in different locations. *Open source software like TOMCAT, JAVA, Mysql and Linux is used to minimize the cost for the Customer.*

# 5. Methodology

**SDLC (Software Development Life Cycle) – Umbrella Model**

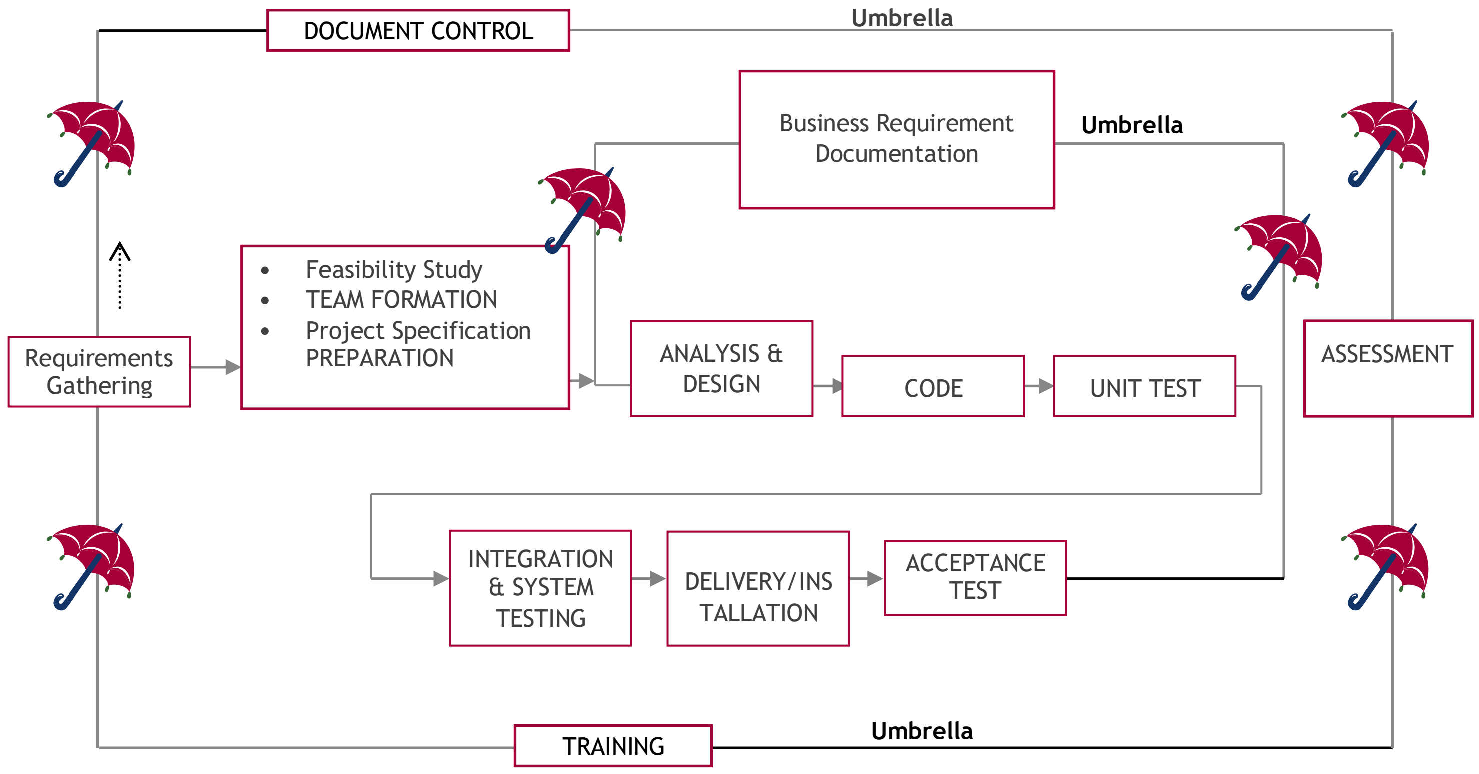


**Fig no. 5.1 Umbrella model**

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

**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.

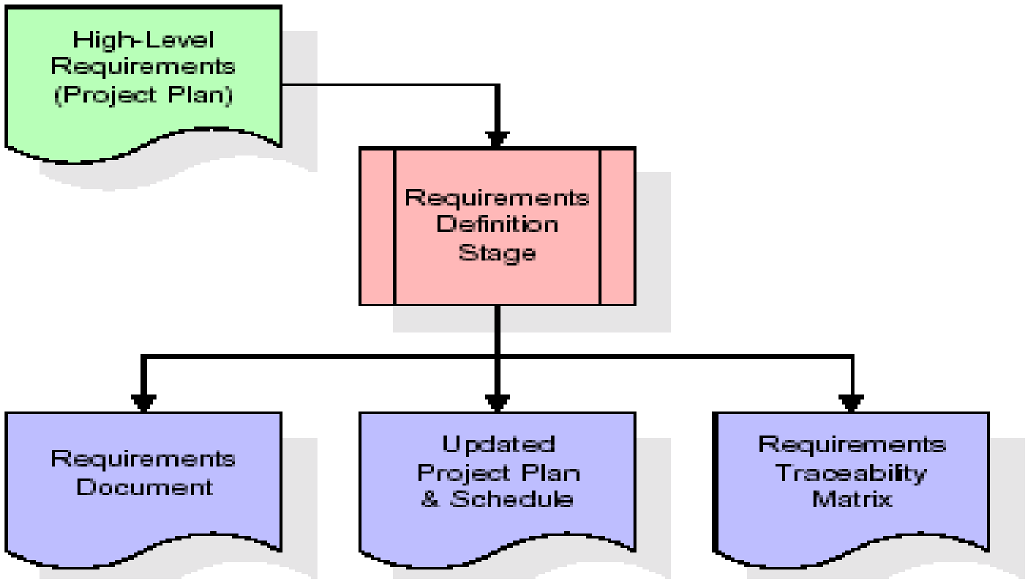

High-Level Requirements (Project Plan)
Requirements Definition Stage
Requirements Document
Updated Project Plan & Schedule
Requirements Traceability Matrix

**Fig no. 5.2 Requirements Gathering stage**

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, number 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.

## 5.3Analysis 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.

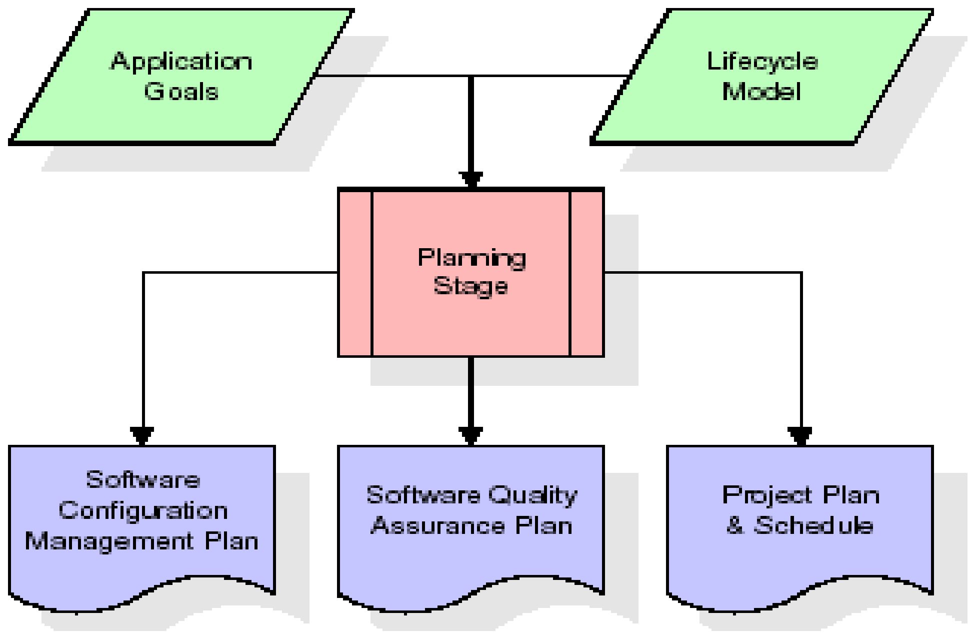

Application Goals
Lifecycle Model
Planning Stage
Software Configuration Management Plan
Software Quality Assurance Plan
Project Plan & Schedule

**Fig no. 5.3 Analysis stage**

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.

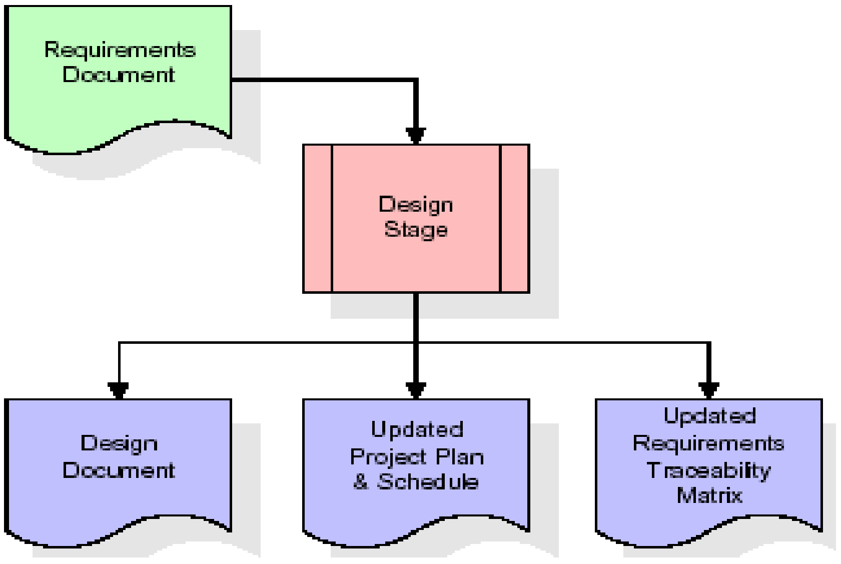

Requirements Document
Design Stage
Design Document
Updated Project Plan & Schedule
Updated Requirements Traceability Matrix

**Fig no. 5.4 Designing stage**

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, 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.

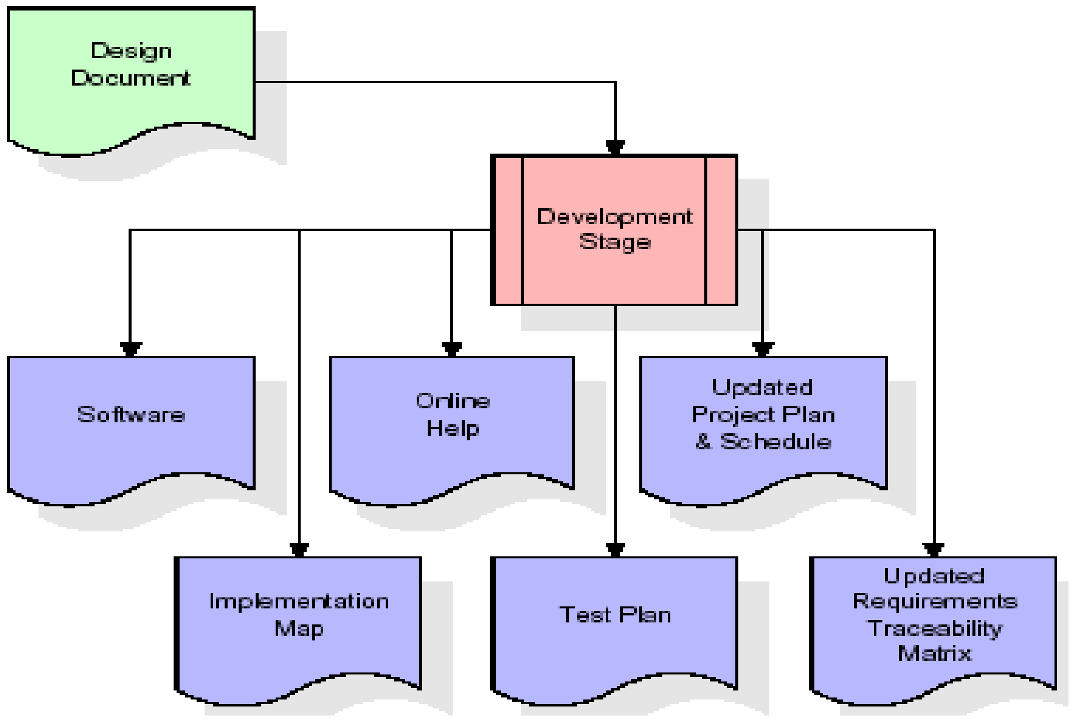


**Fig no. 5.5 Coding stage**

## 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.

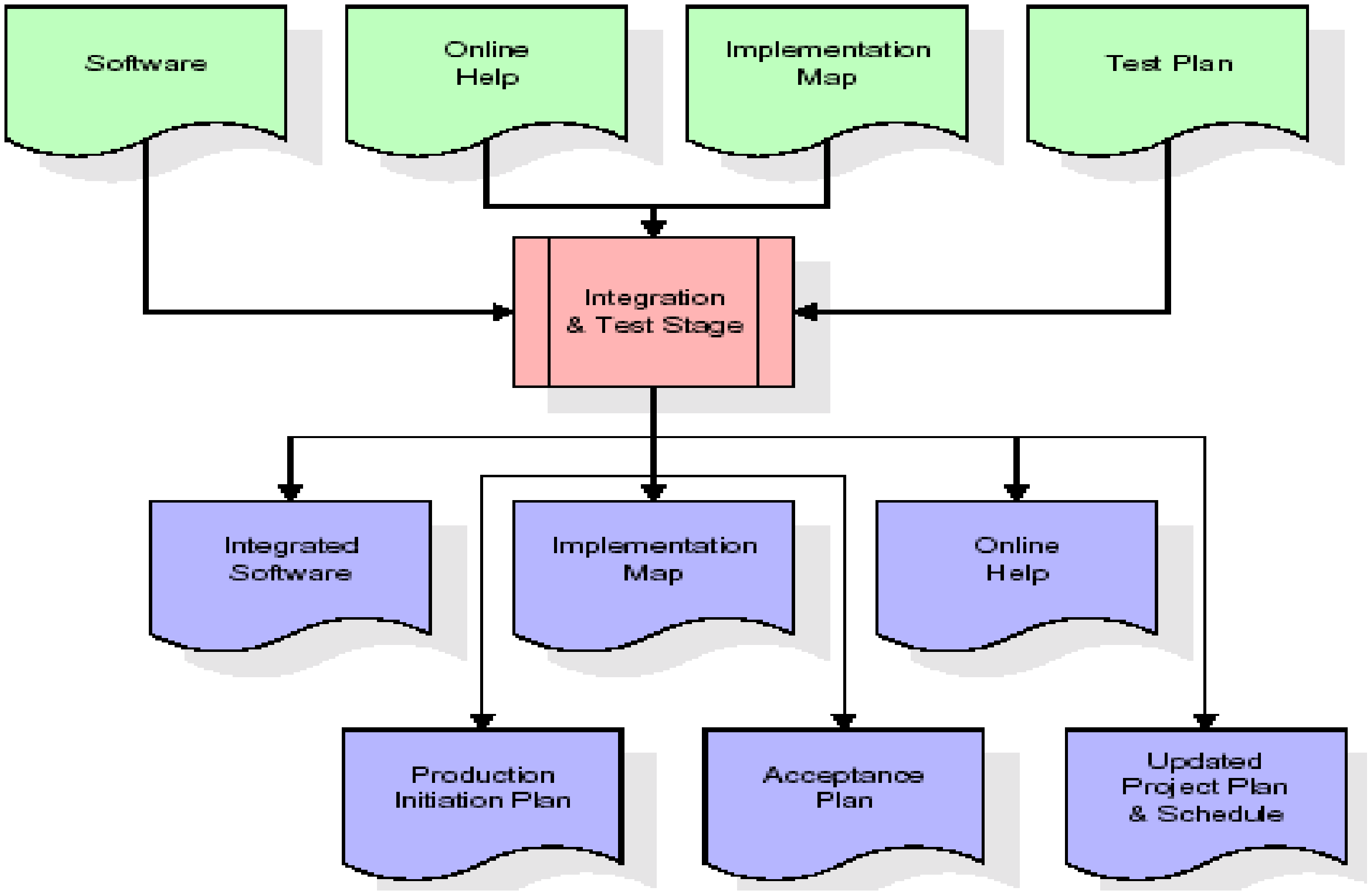


**Fig no. 5.6 Integration and Testing Stage**

## Installation & Acceptance Test

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded 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.

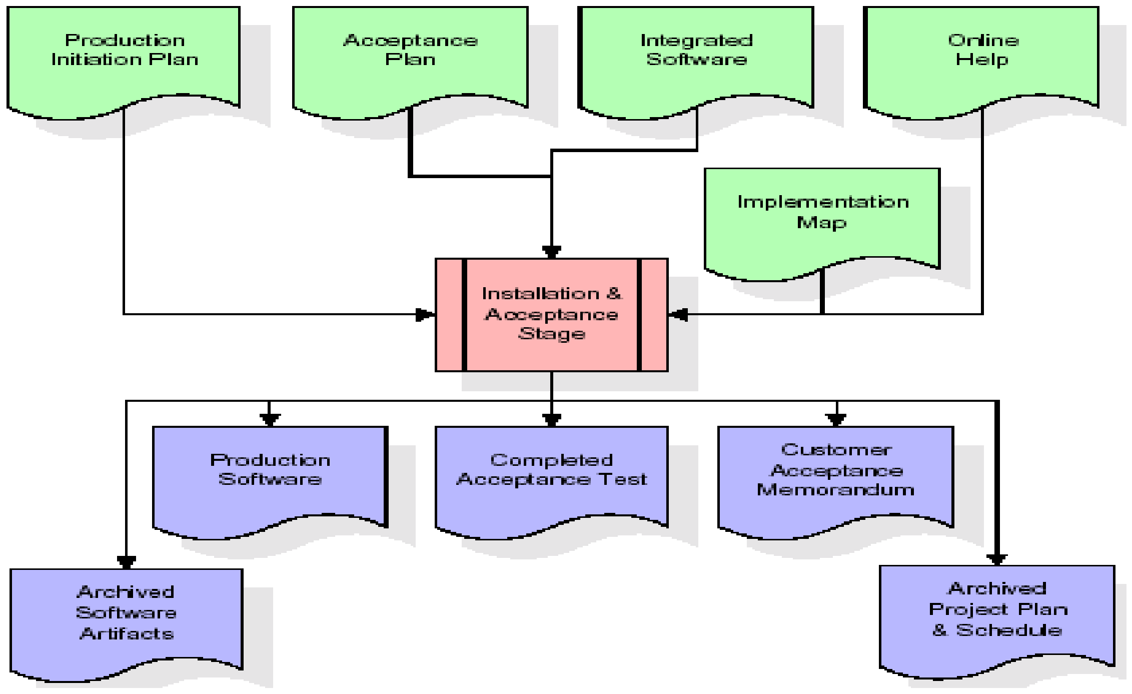

Production Initiation Plan
Acceptance Plan
Integrated Software
Online Help
Implementation Map
Installation & Acceptance Stage
Production Software
Completed Acceptance Test
Customer Acceptance Memorandum
Archived Software Artifacts
Archived Project Plan & Schedule

**Fig no. 5.7 Installation**

### 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.

# 6. System Design

## 6.1 SYSTEM ARCHITECTURE

The purpose of the design phase is to arrange an answer of the matter such as by the necessity document. This part is that the opening moves in moving the matter domain to the answer domain. The design phase satisfies the requirements of the system. The design of a system is probably the foremost crucial issue warm heartedness the standard of the software package. It’s a serious impact on the later part, notably testing and maintenance.

The output of this part is that the style of the document. This document is analogous to a blueprint of answer and is employed later throughout implementation, testing and maintenance. The design activity is commonly divided into 2 separate phases System Design and Detailed Design.

System Design conjointly referred to as top-ranking style aims to spot the modules that ought to be within the system, the specifications of those modules, and the way them move with one another to supply the specified results.

At the top of the system style all the main knowledge structures, file formats, output formats, and also the major modules within the system and their specifications square measure set. System design is that the method or art of process the design, components, modules, interfaces, and knowledge for a system to satisfy such as needs. Users will read it because the application of systems theory to development.

Detailed Design, the inner logic of every of the modules laid out in system design is determined. Throughout this part, the small print of the info of a module square measure sometimes laid out in a high-level style description language that is freelance of the target language within which the software package can eventually be enforced.

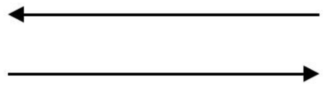
In system design the main target is on distinguishing the modules, whereas throughout careful style the main target is on planning the logic for every of the modules.

## 6.2 DATA FLOW DIAGRAMS

Data Flow Diagram can also be termed as bubble chart. It is a pictorial or graphical form, which can be applied to represent the input data to a system and multiple functions carried out on the data and the generated output by the system.

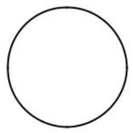
A graphical tool accustomed describe and analyze the instant of knowledge through a system manual or automatic together with the method, stores of knowledge, and delays within the system. The transformation of knowledge from input to output, through processes, is also delineate logically and severally of the physical elements related to the system. The DFD is also known as a data flow graph or a bubble chart.The BasicNotation used to create a DFD's are as follows:

- **Dataflow:**

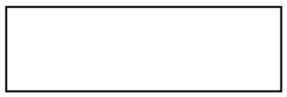


- **Process:**

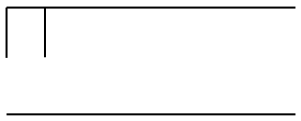
.



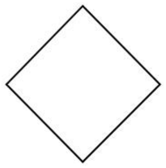
- **Source:**



- **Data Store:**



- **Rhombus**: decision



## 6.3 UML DIAGRAMS

The Unified Modeling Language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic semantic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

### User Model View

This view represents the system from the users perspective. The analysis representation describes a usage scenario from the end-users perspective.

### Structural Model view

In this model the data and functionality are arrived from inside the system. This model view models the static structures.

### Behavioral Model View

It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

### Implementation Model View

In this the structural and behavioral as parts of the system are represented as they are to be built.

## 6.3.1 USE CASE 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.

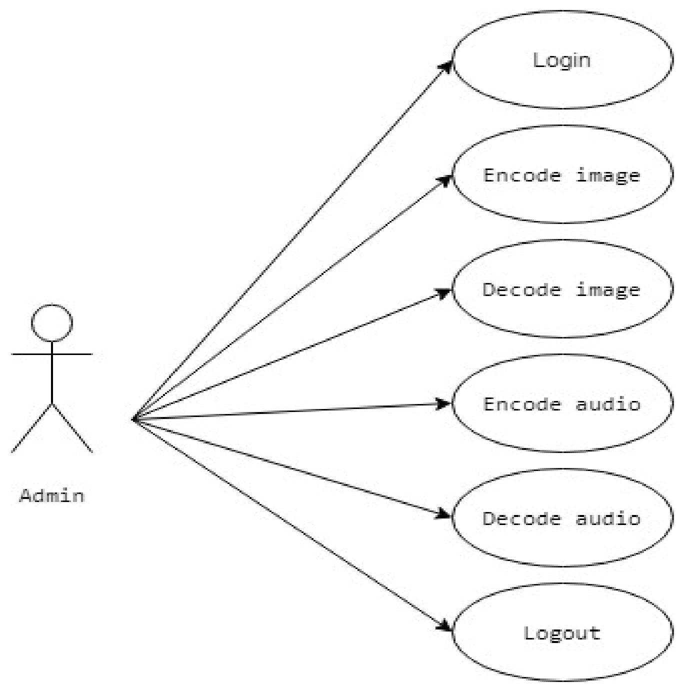


Figure 6.3.1 Use Case Diagram

## 6.3.2 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. A class with three sections, in the diagram, classes is 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.

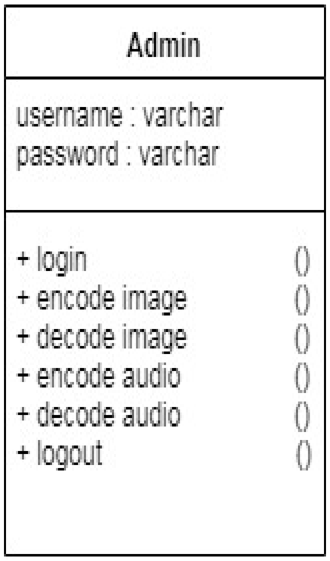


Figure 7.3.2: Class Diagram.

### 6.3.3 SEQUENCEDIAGRAM

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.

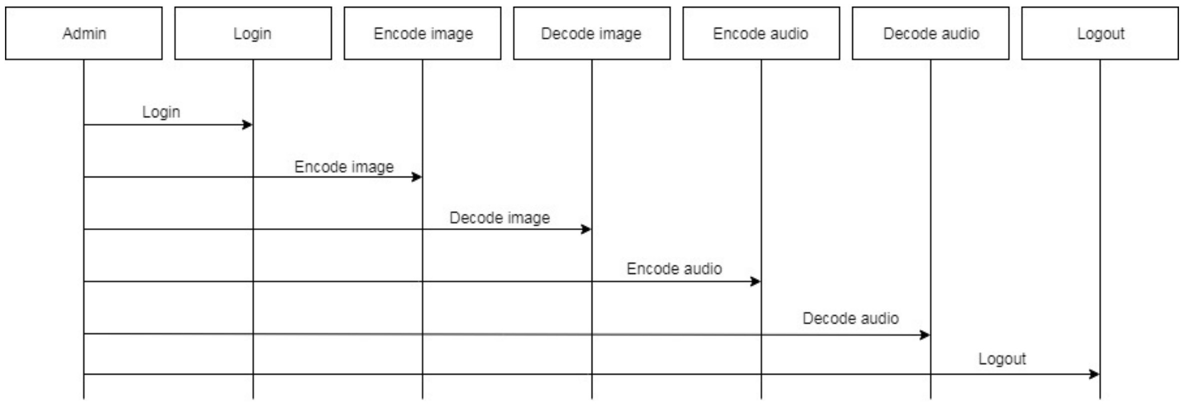


Figure 7.3.3: Sequence diagram

## 6.3.4 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

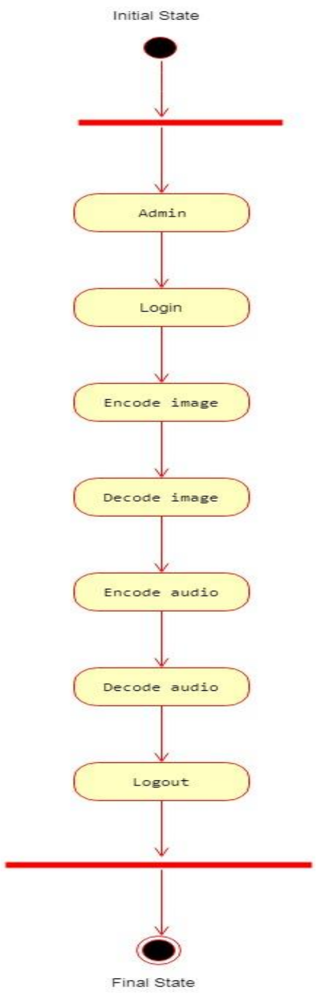


Figure 7.3.4: Activity Diagram

# 7. 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. The following is the description of the testing strategies, which were carried out during the testing period.

## 7.1 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 user 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.

## 7.2 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. Each module in the system is tested separately. In this system the  resource classification and job scheduling modules are tested separately.

## 7.3 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

## 7.4 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.

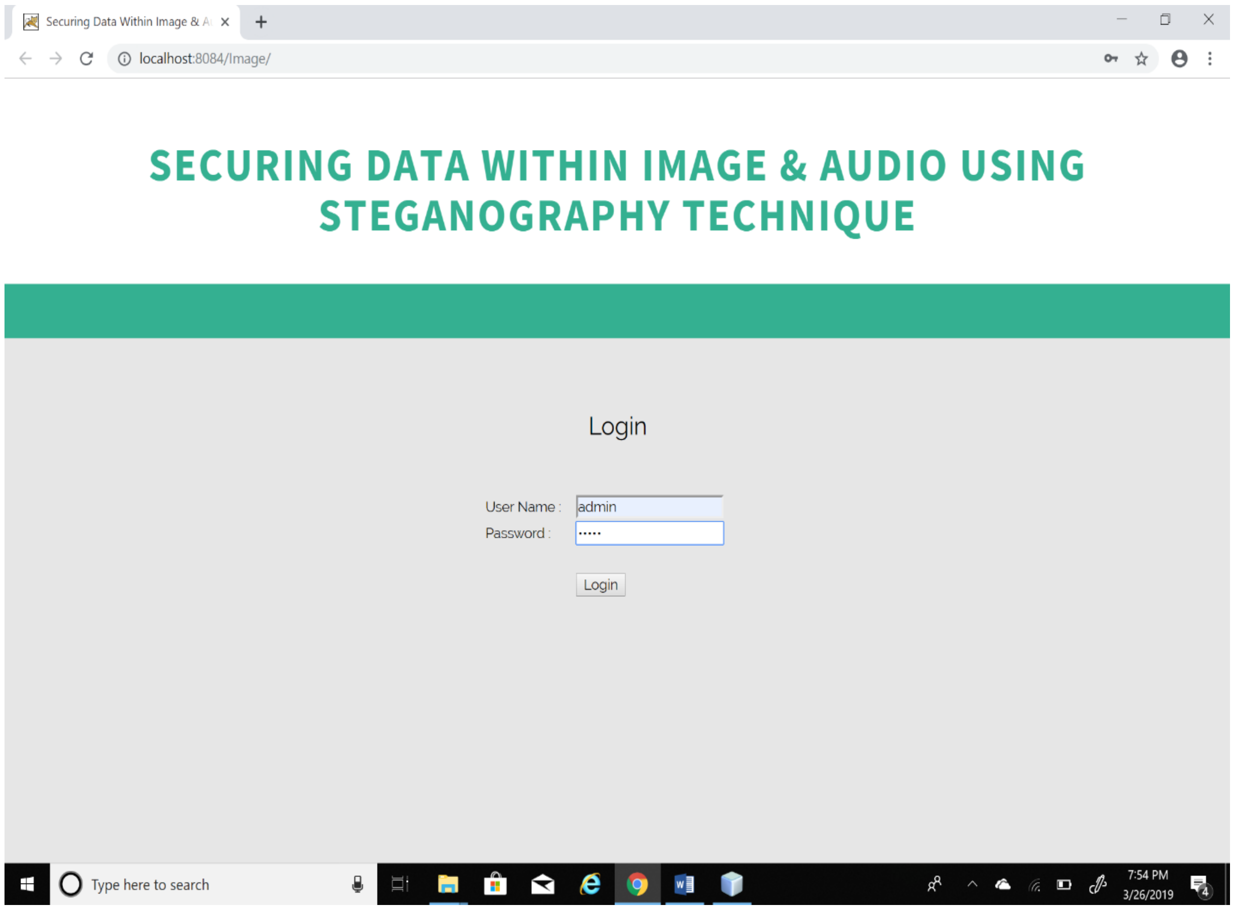
## 7.5 TEST CASES:

| **Test Case Id** | **Test Case Name** | **Test Case Desc.** | **Test Steps** | | | **Test Case Status** | **Test Priority** |
|---|---|---|---|---|---|---|---|
| | | | **Step** | **Expected** | **Actual** | | |
| 01 | Upload the tasks dataset | Verify either file is loaded or not | If dataset is not uploaded | It cannot display the file loaded message | File is loaded which displays task waiting time | High | High |
| 02 | Upload patients dataset | Verify either dataset loaded or not | If dataset is not uploaded | It cannot display dataset reading process completed | It can display dataset reading process completed | low | High |
| 03 | Preprocessing | Whether preprocessing on the dataset applied or not | If not applied | It cannot display the necessary data for further process | It can display the necessary data for further process | Medium | High |

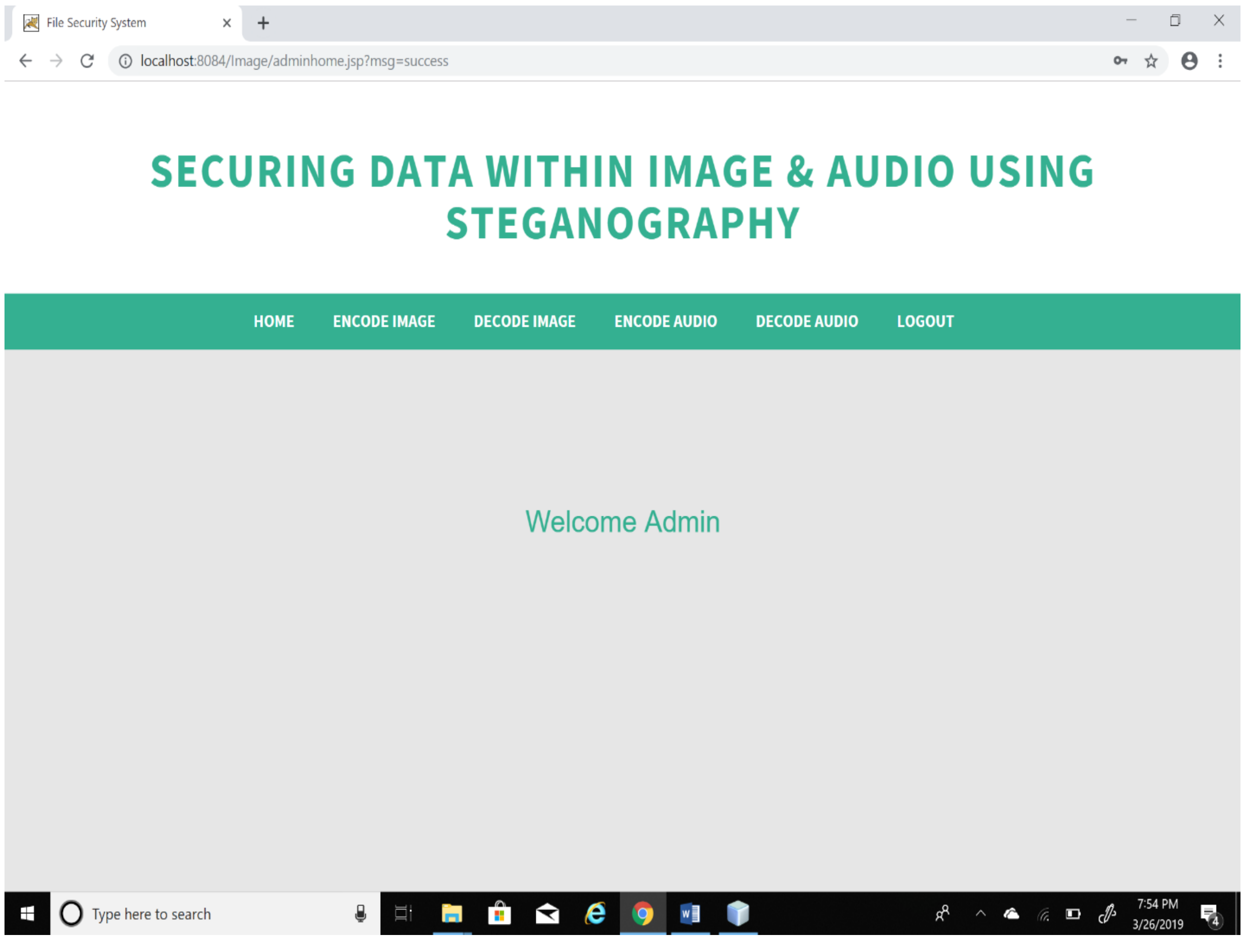
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 04 | Prediction Random Forest | Whether Prediction algorithm applied on the data or not | If not applied | Random tree is not generated | Random tree is generated | High | High |
| 05 | Recommendation | Whether predicted data is displayed or not | If not displayed | It cannot view prediction containing patient data | It can view prediction containing patient data | High | High |
| 06 | Noisy Records Chart | Whether the graph is displayed or not | If graph is not displayed | It does not show the variations in between clean and noisy records | It shows the variations in between clean and noisy records | Low | Medium |

TABLE 7.5.1 TESTCASES

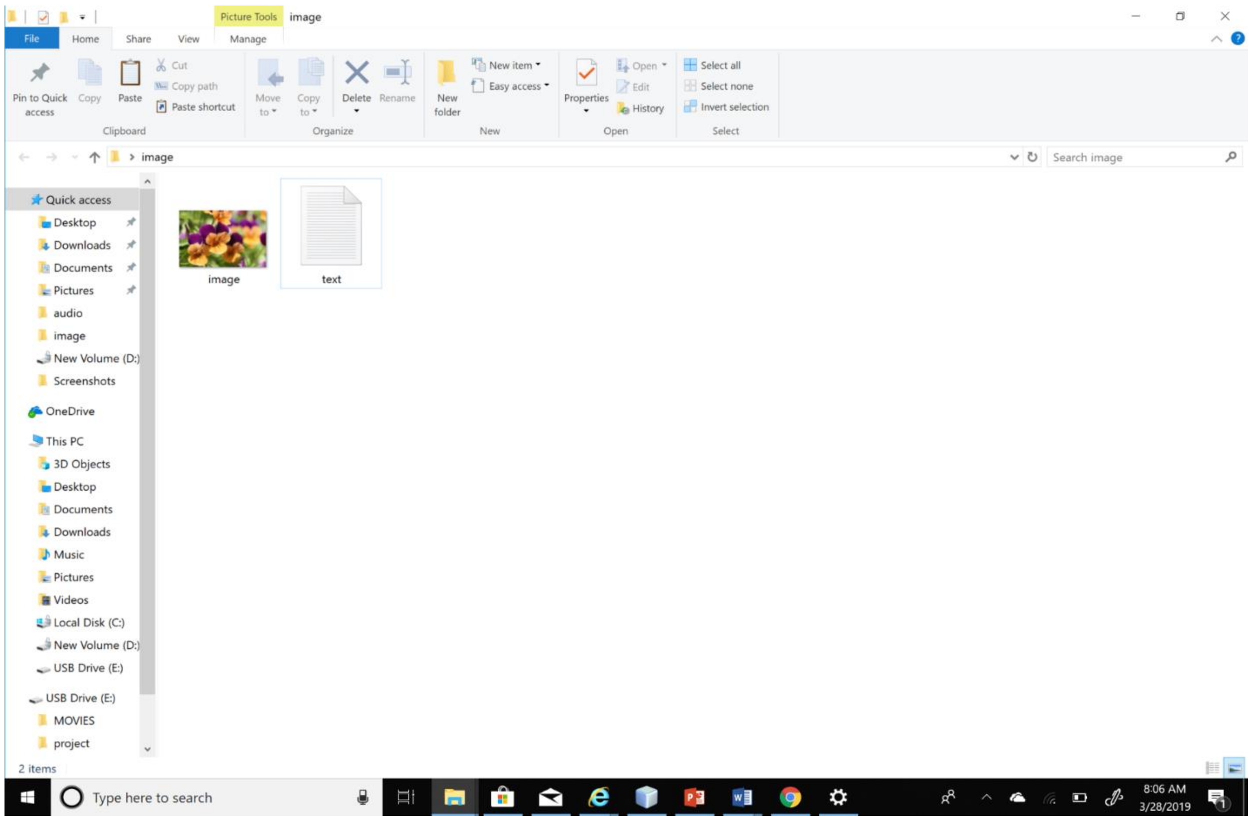
# 8. SCREEN SHOTS

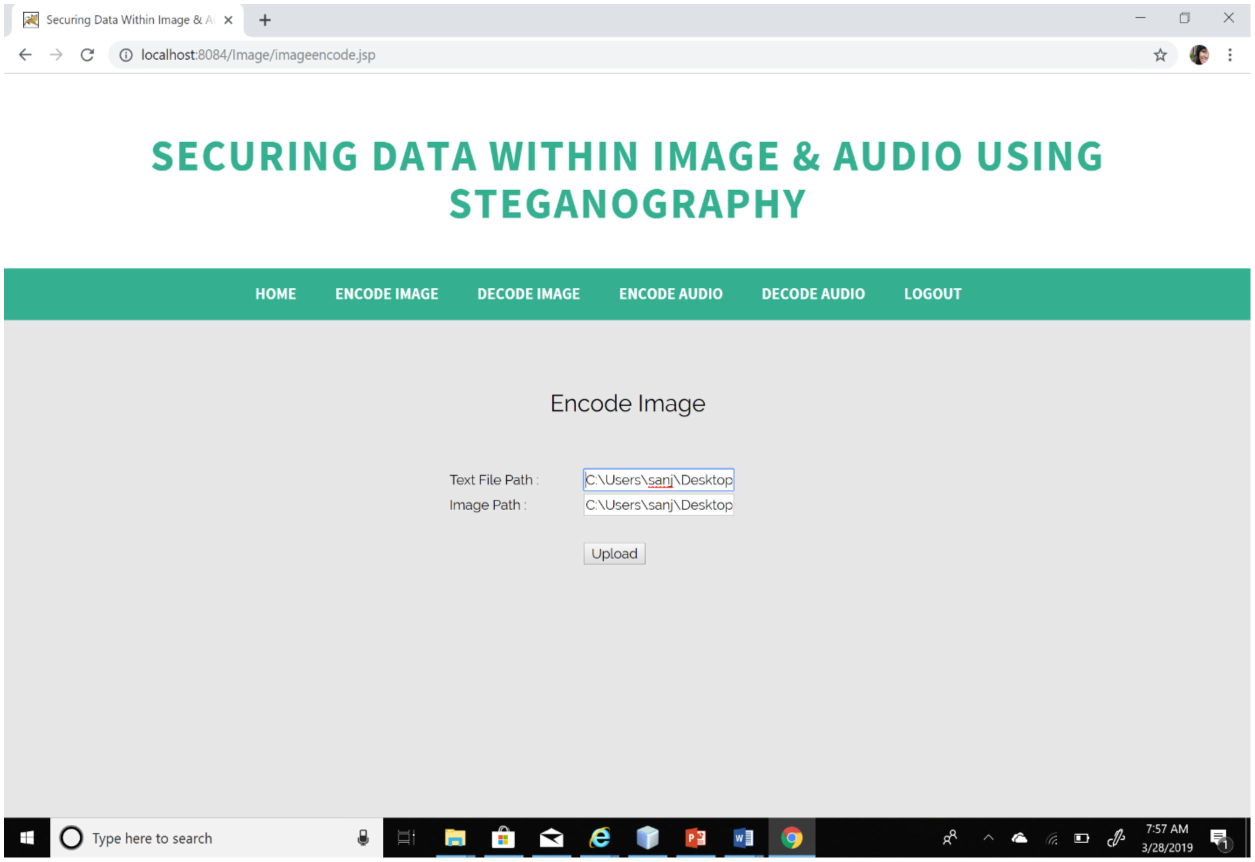

Securing Data Within Image & A
localhost:8084/Image/
SECURING DATA WITHIN IMAGE & AUDIO USING STEGANOGRAPHY TECHNIQUE
Login
Login
Type here to search
7:54 PM
3/26/2019

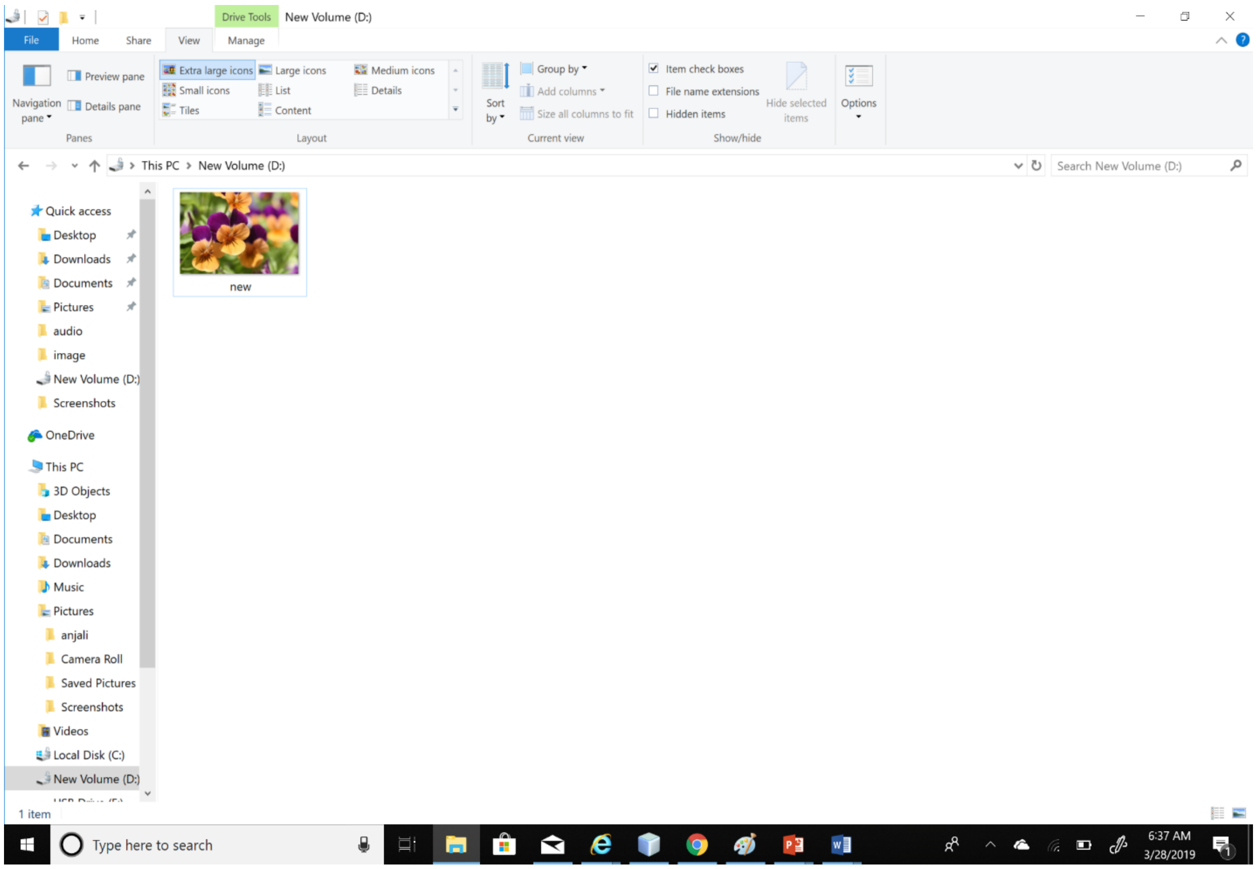
User Name : admin
Password : •••••

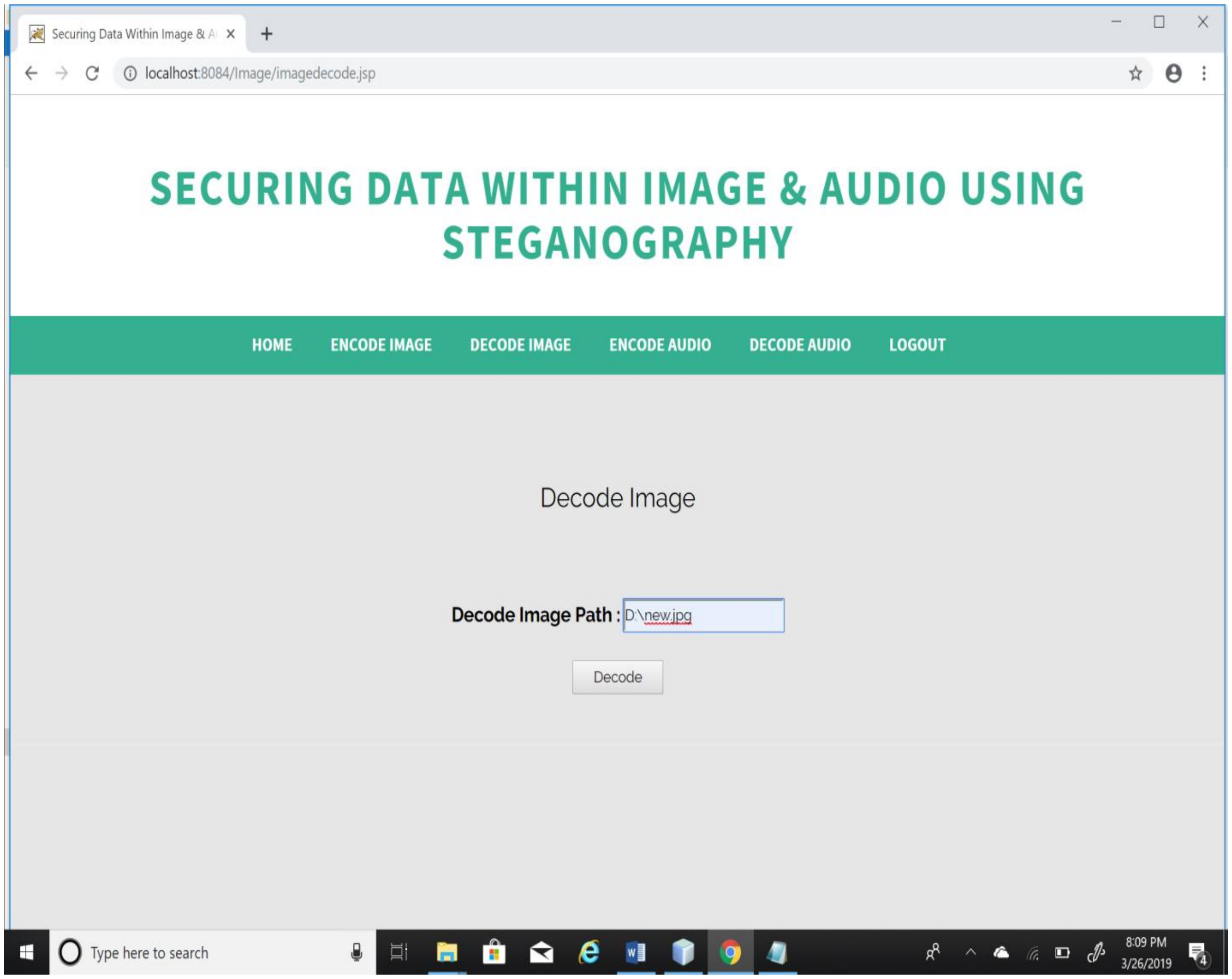

File Security System
localhost:8084/Image/adminhome.jsp?msg=success
SECURING DATA WITHIN IMAGE & AUDIO USING STEGANOGRAPHY
HOME
ENCODE IMAGE
DECODE IMAGE
ENCODE AUDIO
DECODE AUDIO
LOGOUT
Welcome Admin
Type here to search
7:54 PM
3/26/2019

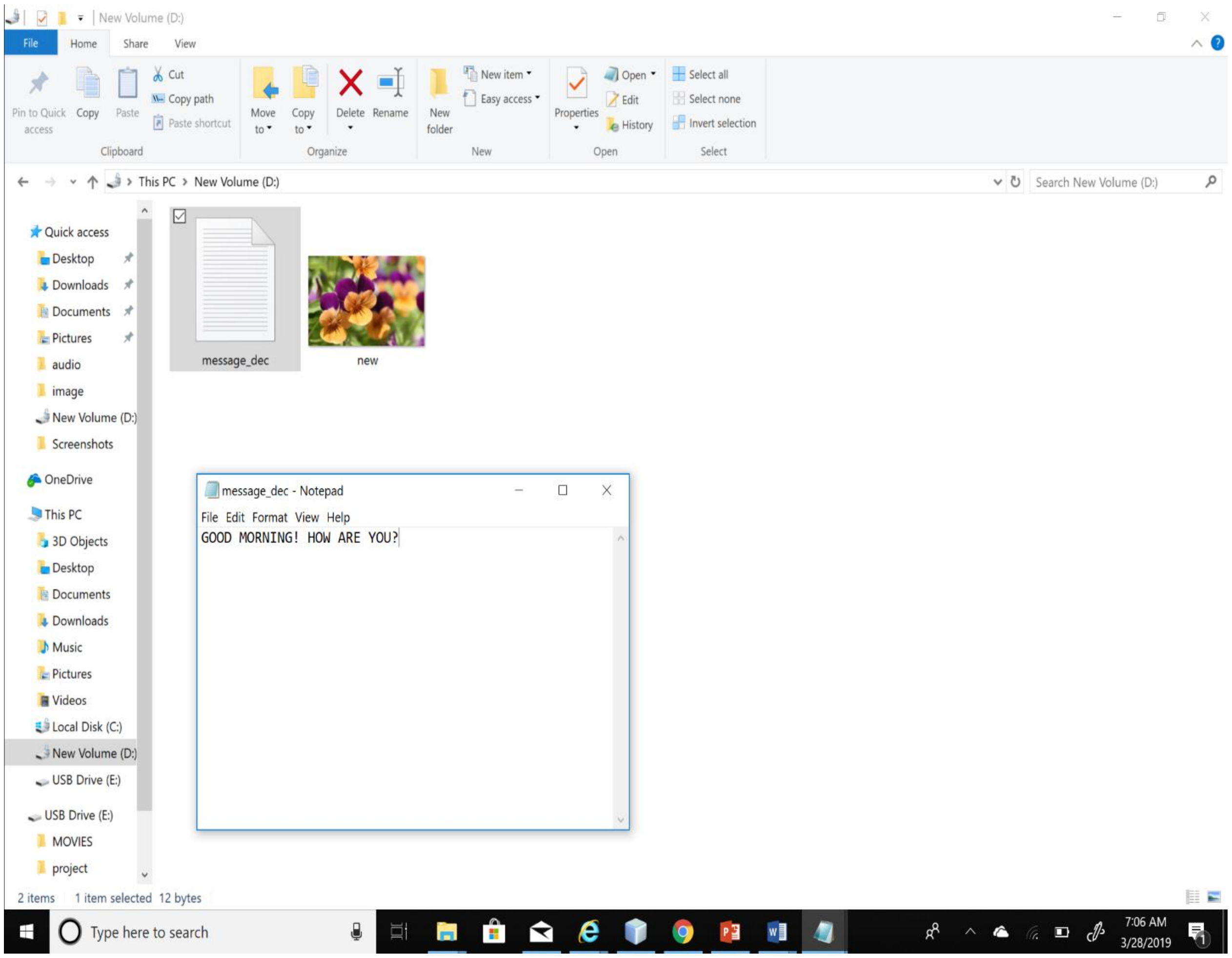
## Create an image and text file with the text that is to be encoded




Securing Data Within Image & A
localhost:8084/Image/imageencode.jsp
SECURING DATA WITHIN IMAGE & AUDIO USING STEGANOGRAPHY
HOME
ENCODE IMAGE
DECODE IMAGE
ENCODE AUDIO
DECODE AUDIO
LOGOUT
Encode Image
Text File Path :
C:\Users\sanj\Desktop
Image Path :
C:\Users\sanj\Desktop
Upload
Type here to search
7:57 AM
3/28/2019

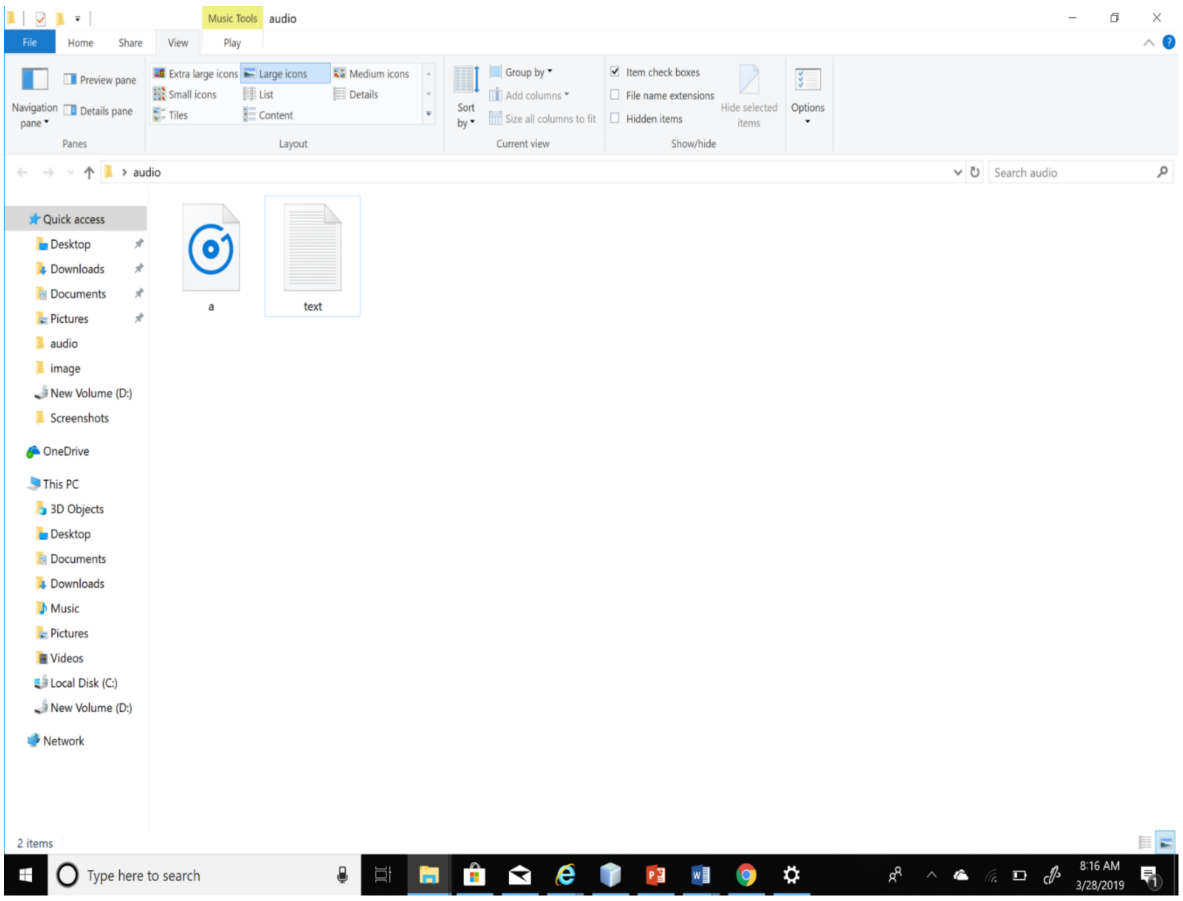

Drive Tools
New Volume (D:)
File
Home
Share
View
Manage
Navigation pane
Preview pane
Details pane
Panes
Extra large icons
Large icons
Medium icons
Small icons
List
Details
Tiles
Content
Layout
Sort by
Group by
Add columns
Size all columns to fit
Current view
Item check boxes
File name extensions
Hidden items
Hide selected items
Show/hide
Options
This PC > New Volume (D:)
Search New Volume (D:)
Quick access
Desktop
Downloads
Documents
Pictures
audio
image
New Volume (D:)
Screenshots
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
anjali
Camera Roll
Saved Pictures
Screenshots
Videos
Local Disk (C:)
New Volume (D:)
new
1 item
Type here to search
6:37 AM
3/28/2019

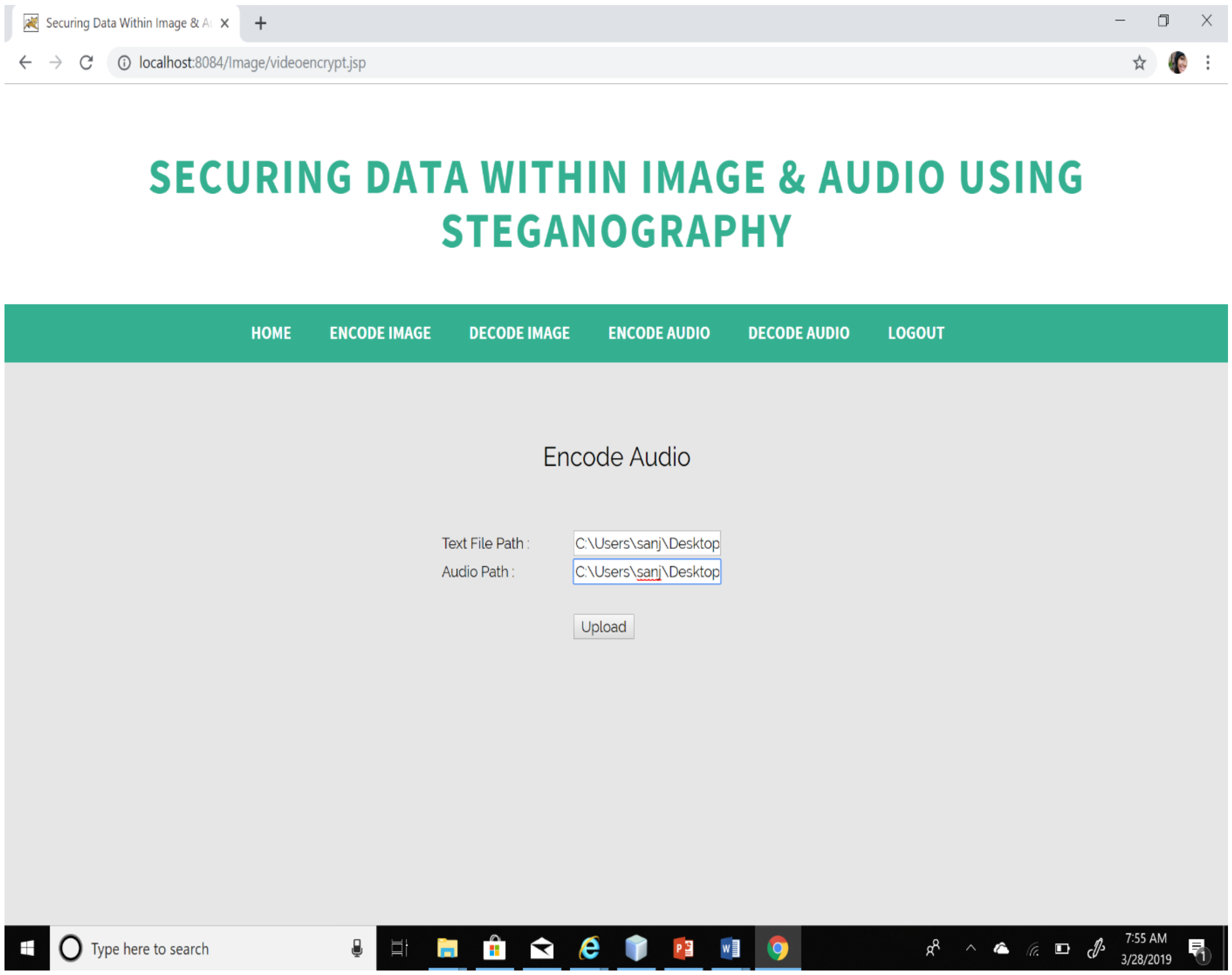

Securing Data Within Image & A
localhost:8084/Image/imagedecode.jsp
SECURING DATA WITHIN IMAGE & AUDIO USING STEGANOGRAPHY
HOME
ENCODE IMAGE
DECODE IMAGE
ENCODE AUDIO
DECODE AUDIO
LOGOUT
Decode Image
Decode Image Path : D:\new.jpg
Decode
Type here to search
8:09 PM
3/26/2019


New Volume (D:)
File
Home
Share
View
This PC > New Volume (D:)
message_dec
new
message_dec - Notepad
File Edit Format View Help
2 items 1 item selected 12 bytes

```
GOOD MORNING! HOW ARE YOU?
```

## Create an audio and text file that is to be encoded

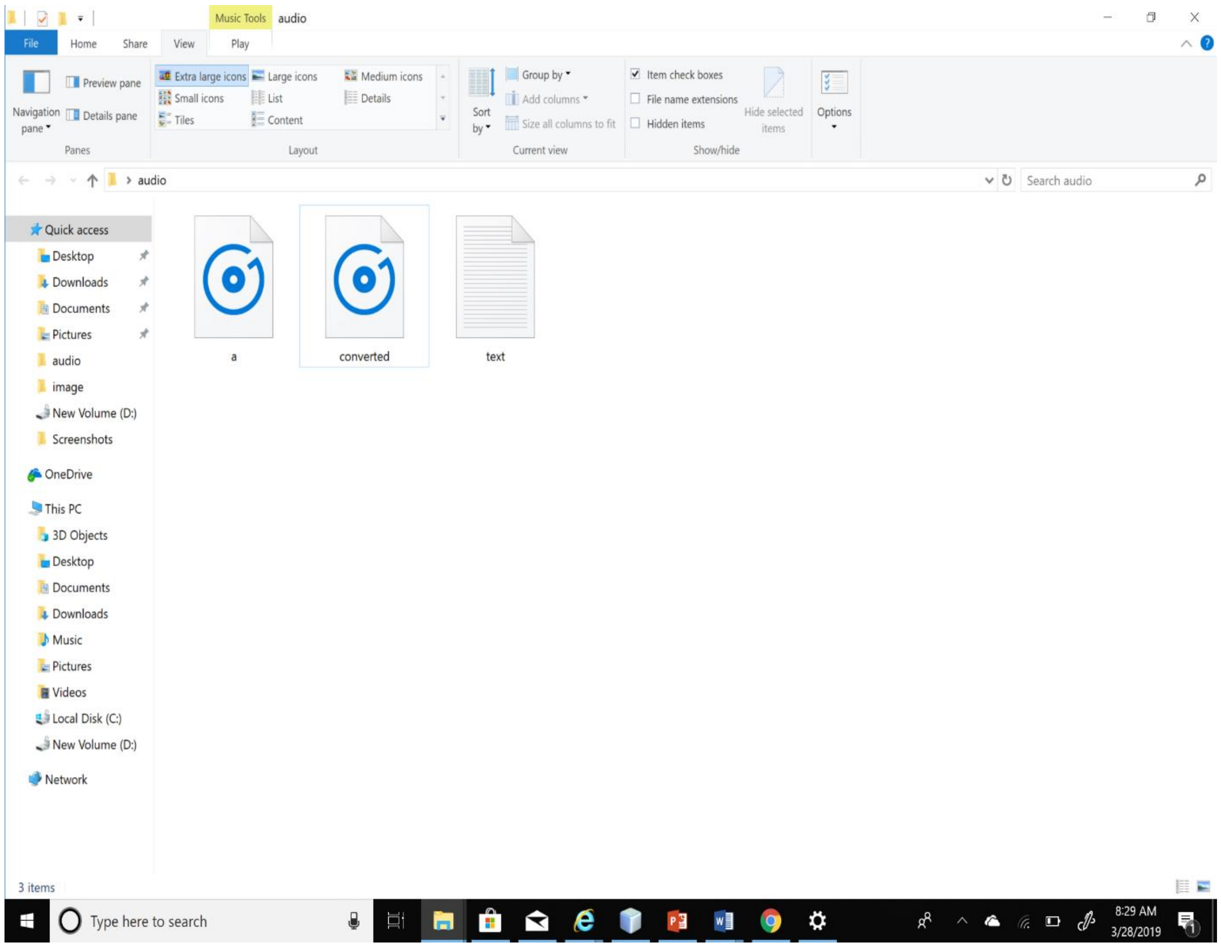


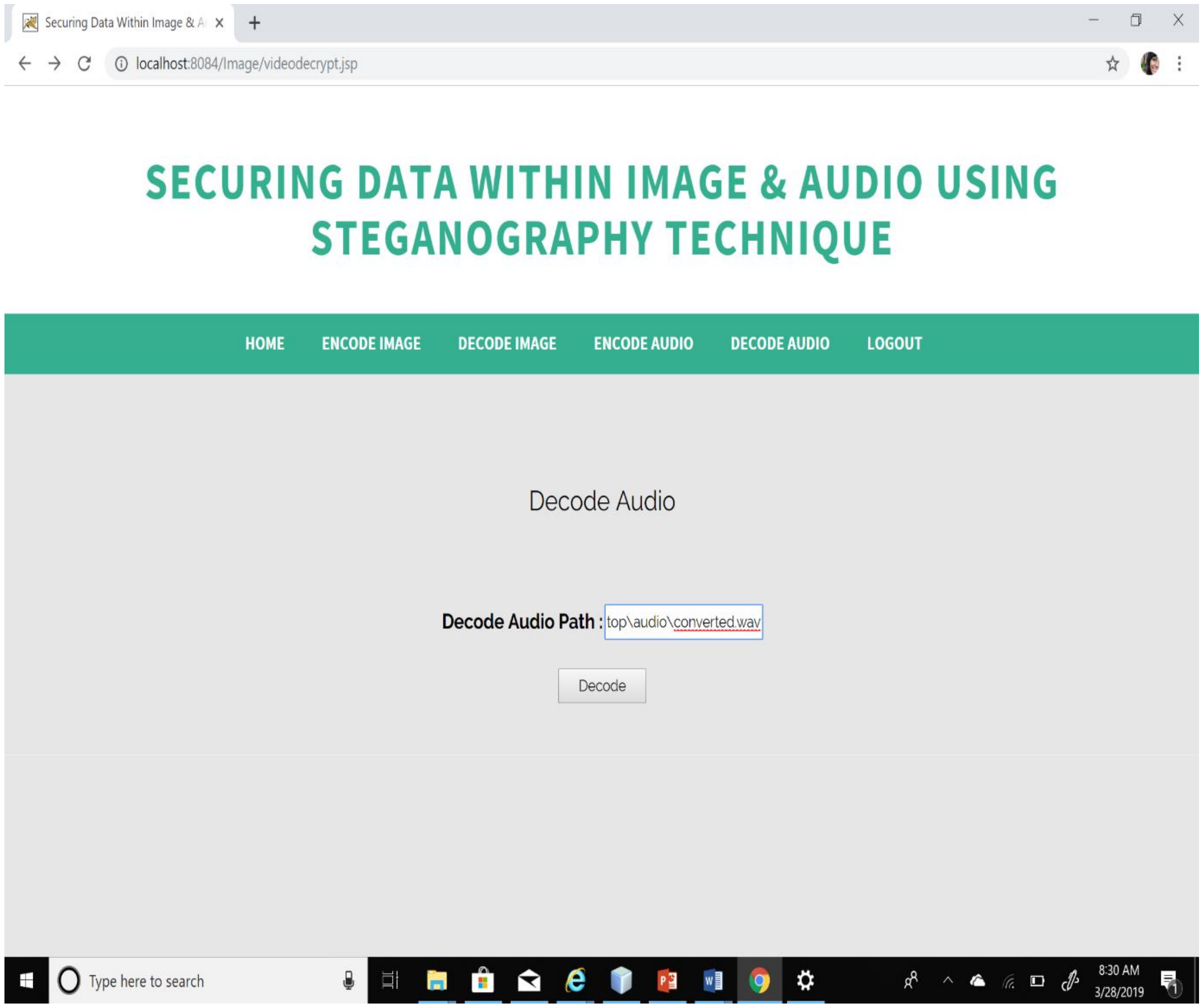

Securing Data Within Image & A
localhost:8084/Image/videoencrypt.jsp

SECURING DATA WITHIN IMAGE & AUDIO USING STEGANOGRAPHY

Encode Audio

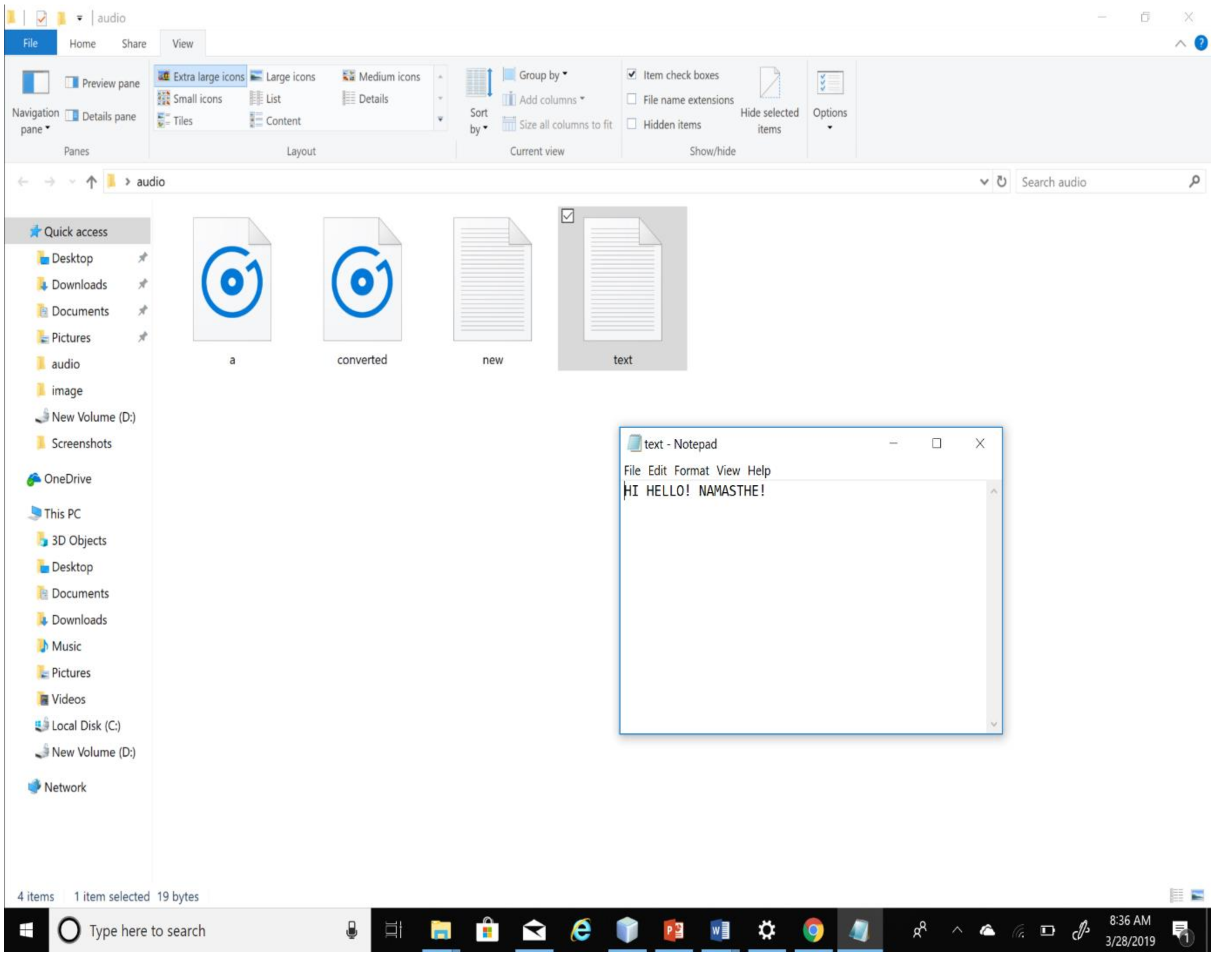
Text File Path : C:\Users\sanj\Desktop
Audio Path : C:\Users\sanj\Desktop


Music Tools
audio
File
Home
Share
View
Play
Navigation pane
Preview pane
Details pane
Panes
Extra large icons
Large icons
Medium icons
Small icons
List
Details
Tiles
Content
Layout
Sort by
Group by
Add columns
Size all columns to fit
Current view
Item check boxes
File name extensions
Hidden items
Hide selected items
Options
Show/hide
audio
Search audio
Quick access
Desktop
Downloads
Documents
Pictures
audio
image
New Volume (D:)
Screenshots
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
New Volume (D:)
Network
a
converted
text
3 items
Type here to search
8:29 AM
3/28/2019


Securing Data Within Image & A
localhost:8084/Image/videodecrypt.jsp
SECURING DATA WITHIN IMAGE & AUDIO USING STEGANOGRAPHY TECHNIQUE
HOME
ENCODE IMAGE
DECODE IMAGE
ENCODE AUDIO
DECODE AUDIO
LOGOUT
Decode
Type here to search
8:30 AM
3/28/2019

Decode Audio

Decode Audio Path : top\audio\converted.wav



```
HI HELLO! NAMASTHE!
```

# 9. CONCLUSION

In This Project we have successfully hide the data in Image and also data hiding in Audio Files.

**Future Enhancements:**

It is not possible to develop a system that makes all the requirements of the user. User requirements keep changing as the system is being used. Some of the future enhancements that can be done to this system are:

- As the technology emerges, it is possible to upgrade the system and can be adaptable to desired environment.
- Based on the future security issues, security can be improved using emerging technologies like single sign-on.

# 10. REFERENCES

[1] I.J. Cox, M.L. Miller, J.A. Bloom, Digital watermarking, Morgan Kaufmann, 2001.

[2] Mohannad Ahmad AbdulAziz Al-Dharrab," Benchmarking Framework for Software Watermarking" King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia, June 2005.

[3] J. Nagra, C. Thomborson, and C. Collberg, (2002), A functional taxonomy for software watermark- ing, in M. Oudshoorn, ed., `Proc. 25th Australasian Computer Science Conference 2002', ACS, pp. 177-186.

[4] Ersin Elbasi and Ahmet M. Eskicioglu," A SEMI-BLIND WATERMARKING SCHEME FOR IMAGES USING A TREE STRUCTURE", Sarnoff Symposium, 2006 IEEE

[5] Saeid Fazli and Gholamreza Khodaverdi, "Trade-off between Imperceptibility and Robustness of LSB Watermarking using SSIM Quality Metrics", 978-0-7695-3944-7/10 $26.00 © 2010 IEEE DOI 10.1109/ICMV.2009.68

[6] Gil-Je Lee, Eun-Jun Yoon, Kee-Young Yoo, "A new LSB based DigitalWatermarking Scheme with Random Mapping Function", 978-0-7695-3427-5/08 $25.00 © 2008 IEEE DOI 10.1109/UMC.2008.33

[7] Gaurav Bhatnagar, Balasubramanian Raman," A new robust reference watermarking scheme based on DWT-SVD", 0920- 5489/$ – see front matter © 2008 Elsevier B.V. All rights reserved. doi:10.1016/j.csi.2008.09.031

## BIBLIOGRAPHY

JAVA Technologies

JAVA Complete Reference

Java Script Programming by Yehuda Shiran

Mastering JAVA Security

J2EE Professional by Shadab siddiqui

JAVA server pages by Larne Pekowsley

JAVA Server pages by Nick Todd

HTML

HTML Black Book by Holzner

JDBC

Java Database Programming with JDBC by Patel moss.

Software Engineering by Roger Pressman

## SAMPLE CODE

### DB Connection Code:

```
package databasecon;

import java.sql.Connection;
import java.sql.DriverManager;

public class Dbconnection {

    public static Connection getConnection() {
        Connection con = null;
        try {
            Class.forName("com.mysql.jdbc.Driver");
            con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/studentpe
rformance", "root", "root");
        } catch (Exception ex) {
            ex.printStackTrace();
        }
        return con;
    }
}
```

**LSB Decode Code:**

```
package lsb;


import java.awt.image.BufferedImage;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintStream;
import java.io.PrintWriter;
//import java.util.Arrays;
import java.util.Scanner;

import javax.imageio.ImageIO;
public class LSB_decode {
      //static final String STEGIMAGEFILE = "D:\\2.jpg";
      //static final String DECODEDMESSAGEFILE =
"D:\\message_dec.txt";

      public static String b_msg="";
      public static int len = 0;




       public void decode(String STEGIMAGEFILE, String
DECODEDMESSAGEFILE) {

             try{
            BufferedImage yImage=readImageFile(STEGIMAGEFILE);

DecodeTheMessage(yImage);
String msg="";
//System.out.println("len is "+len*8);
for(int i=0;i<len*8;i=i+8){

      String sub=b_msg.substring(i,i+8);

      int m=Integer.parseInt(sub,2);
      char ch=(char) m;
      System.out.println("m "+m+" c "+ch);
      msg+=ch;
}
PrintWriter out = new PrintWriter(new
FileWriter(DECODEDMESSAGEFILE, true), true);
out.write(msg);
out.close();

 } catch (Exception e) {
            System.out.println(e);
        }
}
```

```
public static BufferedImage readImageFile(String COVERIMAGEFILE){
BufferedImage theImage = null;
File p = new File (COVERIMAGEFILE);
try{
theImage = ImageIO.read(p);
}catch (IOException e){
e.printStackTrace();
System.exit(1);
}
return theImage;
}


public static void DecodeTheMessage (BufferedImage yImage) throws
Exception{

int j=0;
int currentBitEntry=0;
String bx_msg="";
for (int x = 0; x < yImage.getWidth(); x++){
for ( int y = 0; y < yImage.getHeight(); y++){
if(x==0&&y<8){
      //System.out.println("enc "+yImage.getRGB(x, y)+" dec
"+yImage.getRGB(x, y)+" "+b_msg);
      int currentPixel = yImage.getRGB(x, y);
      int red = currentPixel>>16;
      red = red & 255;
      int green = currentPixel>>8;
      green = green & 255;
      int blue = currentPixel;
      blue = blue & 255;
      String x_s=Integer.toBinaryString(blue);
      bx_msg+=x_s.charAt(x_s.length()-1);
      len=Integer.parseInt(bx_msg,2);

}
else if(currentBitEntry<len*8){
//System.out.println("enc "+yImage.getRGB(x, y)+" dec
"+yImage.getRGB(x, y)+" "+b_msg);
      int currentPixel = yImage.getRGB(x, y);
      int red = currentPixel>>16;
      red = red & 255;
      int green = currentPixel>>8;
      green = green & 255;
      int blue = currentPixel;
      blue = blue & 255;
      String x_s=Integer.toBinaryString(blue);
      b_msg+=x_s.charAt(x_s.length()-1);
      currentBitEntry++;
      //System.out.println("curre "+currentBitEntry);
}
}
}
System.out.println("bin value of msg hided in img is "+b_msg);
}
}
```

**Encode Code:**

```
package lsb;

import java.awt.image.BufferedImage;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
//import java.util.Arrays;
import java.util.Scanner;

import javax.imageio.ImageIO;
public class LSB_encode {
      //static final String MESSAGEFILE = "D:\\message.txt";
      //static final String COVERIMAGEFILE = "D:\\Desert.jpg";
      //static final String STEGIMAGEFILE = "D:\\2.jpg";


        public void encode(String MESSAGEFILE, String
COVERIMAGEFILE,String STEGIMAGEFILE) {

           try{
           String contentOfMessageFile =
(readMessageFile(MESSAGEFILE));
int[] bits=bit_Msg(contentOfMessageFile);
System.out.println("msg in file "+contentOfMessageFile);
    for(int i=0;i<bits.length;i++)
System.out.print(bits[i]);
System.out.println();
BufferedImage theImage=readImageFile(COVERIMAGEFILE);
hideTheMessage(bits, theImage,STEGIMAGEFILE);
 } catch (Exception e) {
            System.out.println(e);
        }
}

public static String readMessageFile (String MESSAGEFILE) throws
FileNotFoundException{
     String contentOfMessageFile = "";
     File a = new File (MESSAGEFILE);
     Scanner scan = new Scanner (a);
     while (scan.hasNextLine()){
     String next = scan.nextLine();
     contentOfMessageFile += next;
     if (scan.hasNextLine()){
     contentOfMessageFile += "\n";
     }
     }
     scan.close();
     return contentOfMessageFile;
     }
public static int[] bit_Msg(String msg){
     int j=0;
     int[] b_msg=new int[msg.length()*8];
     for(int i=0;i<msg.length();i++){
          int x=msg.charAt(i);
          String x_s=Integer.toBinaryString(x);
          while(x_s.length()!=8){
               x_s='0'+x_s;
          }
          System.out.println("dec value for "+x +" is "+x_s);

          for(int i1=0;i1<8;i1++) {
              b_msg[j] =
Integer.parseInt(String.valueOf(x_s.charAt(i1)));
              j++;
            };
     }

     return b_msg;
}
public static BufferedImage readImageFile(String COVERIMAGEFILE){
BufferedImage theImage = null;
File p = new File (COVERIMAGEFILE);
try{
theImage = ImageIO.read(p);
}catch (IOException e){
e.printStackTrace();
System.exit(1);
}
return theImage;
}


public static void hideTheMessage (int[] bits, BufferedImage
theImage,String STEGIMAGEFILE) throws Exception{
     File f = new File (STEGIMAGEFILE);
```

```
      BufferedImage sten_img=null;
      int bit_l=bits.length/8;
      int[] bl_msg=new int[8];
      System.out.println("bit lent "+bit_l);
      String bl_s=Integer.toBinaryString(bit_l);
      while(bl_s.length()!=8){
            bl_s='0'+bl_s;
      }
      for(int i1=0;i1<8;i1++) {
            bl_msg[i1] =
Integer.parseInt(String.valueOf(bl_s.charAt(i1)));
        };
int j=0;
int b=0;
int currentBitEntry=8;

for (int x = 0; x < theImage.getWidth(); x++){
for ( int y = 0; y < theImage.getHeight(); y++){
      if(x==0&&y<8){
            int currentPixel = theImage.getRGB(x, y);
            int ori=currentPixel;
            int red = currentPixel>>16;
            red = red & 255;
            int green = currentPixel>>8;
            green = green & 255;
            int blue = currentPixel;
            blue = blue & 255;
            String x_s=Integer.toBinaryString(blue);
            String sten_s=x_s.substring(0, x_s.length()-1);
            sten_s=sten_s+Integer.toString(bl_msg[b]);

            //j++;
            int temp=Integer.parseInt(sten_s,2);
            int s_pixel=Integer.parseInt(sten_s, 2);
            int a=255;
            int rgb = (a<<24) | (red<<16) | (green<<8) | s_pixel;
            theImage.setRGB(x, y, rgb);
            //System.out.println("original "+ori+" after
"+theImage.getRGB(x, y));
            ImageIO.write(theImage, "png", f);
b++;

      }
      else if (currentBitEntry < bits.length+8 ){

      int currentPixel = theImage.getRGB(x, y);
      int ori=currentPixel;
      int red = currentPixel>>16;
      red = red & 255;
      int green = currentPixel>>8;
      green = green & 255;
      int blue = currentPixel;
      blue = blue & 255;
      String x_s=Integer.toBinaryString(blue);
      String sten_s=x_s.substring(0, x_s.length()-1);
      sten_s=sten_s+Integer.toString(bits[j]);
      j++;
      int temp=Integer.parseInt(sten_s,2);
      int s_pixel=Integer.parseInt(sten_s, 2);

      int a=255;
      int rgb = (a<<24) | (red<<16) | (green<<8) | s_pixel;
      theImage.setRGB(x, y, rgb);
      //System.out.println("original "+ori+" after
"+theImage.getRGB(x, y));
      ImageIO.write(theImage, "png", f);

      currentBitEntry++;
      //System.out.println("curre "+currentBitEntry);
      }
}
}
}
}
```