Software Architecture Document for

Security Package

Version 1.0

Prepared by

20z204 - Adarsh G
20z220 - Jeevan Krishna K V
20z259 - Viraj Agarwal
20z267 - Nirmal M
21z431 - Ajay Deepak P M
21z435 - SundarSree B G

Date created:20-09-2022

Revision History

Name	Date	Reason For Changes	Version

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this project is to develop a new approach to hiding secret information in an image, by taking advantage of the benefits of combining cryptography and steganography.

1.2 Document Conventions

In this SRS, font type 'Times New Roman' bold with font size '18' indicates the main headings, font type 'Times New Roman' bold with font size '14' indicates the sub headings and the entire description of each is given by the font type 'Times New Roman' italicized with the font size of '12'.

1.3 Intended Audience and Reading Suggestions

The different types of readers that the document intended for are the developers, integrators of all modules, testers, users(the tutor and others) etc… This SRS contains the overall description, System features, External interface requirements, Non-functional requirements and other requirements. It also has a table of contents with respective page numbers helping the readers to move to the respective pages.

1.4 Product Scope

The Scope of this project is to provide security to the data of the users. For some of the users the data might be changed by the unauthorized person in the network. Only the Authorized persons i.e., who are using our application can change the information. The scope of the project is to hide the data in an image using steganography, provide ciphers and ensure that the quality of concealing data must not be lost.

1.5 References

Books :

- Software Requirements and Specifications: A Lexicon of Practice, Principles and Prejudices (ACM Press) by Michael Jackson
- Software Requirements (Microsoft) Second EditionBy Karl E. Wieger.
- Software Engineering: A Practitioner's Approach Fifth Edition By Roger S. Pressman.

2. Architectural Representation

This document details the architecture using the views defined in the “4+1” model [KRU41], but using the RUP naming convention. The views used to document the security package application are:

Logical view

Audience: Designers.
Area: Functional Requirements: describes the design's object model. Also describes the most important use-case realizations.
Related Artifacts: Design model

Process view

Audience: Integrators.
Area: Non-functional requirements: describes the design's concurrency and synchronization aspects.
Related Artifacts: (no specific artifact).

Implementation view

Audience: Programmers.
Area: Software components: describes the layers and subsystems of the application.
Related Artifacts: Implementation model, components

Deployment view

Audience: Deployment managers.
Area: Topology: describes the mapping of the software onto the hardware and shows the system's distributed aspects.
Related Artifacts: Deployment model.

Use Case view

Audience: all the stakeholders of the system, including the end-users.
Area: describes the set of scenarios and/or use cases that represent some significant, central functionality of the system.
Related Artifacts : Use-Case Model, Use-Case documents

Data view (optional)

Audience: Data specialists, Database administrators
Area: Persistence: describes the architecturally significant persistent elements in the data model
Related Artifacts: Data model.

3. Architectural Goals and Constraints

This section describes the software requirements and objectives that have some significant impact on the architecture

3.1 Technical Platform

The package will be uploaded to PyPi and the Users can install it using pip.

3.2 Security

The package allows users to encrypt and decrypt files before sending them over a public/unknown networks.This means that attackers must not be able to track the algorithms or keys used to decrypt the data being encrypted.

The system ensures that leakage of key/algorithms cannot take place at the same time decrypt the file whenever the key is right.

3.3 Reliability/Availability (failover)

The package is available 24/7 as there are no databases involved in the implementation of the system.

3.4 Internationalization (i18n)

The final package will contain descriptions of all the functions , information of parameters used by each function and the return type.The package will undergo several changes before being uploaded to PyPi (Internationalization) and will be available for installation.

4. Performance

- Time taken to encrypt and decrypt the data : less that 10 seconds required
- The data are encrypted and decrypted using self designed algorithms.

5. Quality

As far as the security package is concerned, the following quality goals have been identified:

Scalability:

- **Description** : The ability of a computer application to continue to function well when it is changed in size or volume in order to meet a user need

- **Solution** : Solution is to cache pre-computed values and optimize queries to the server

Reliability, **Availability**:

- **Description** : Reliability is the measure of how long a machine performs its intended function, whereas availability is the measure of the percentage of time a machine is operable
- **Solution :** Redundancy is a common approach to improve the reliability and availability of a system.

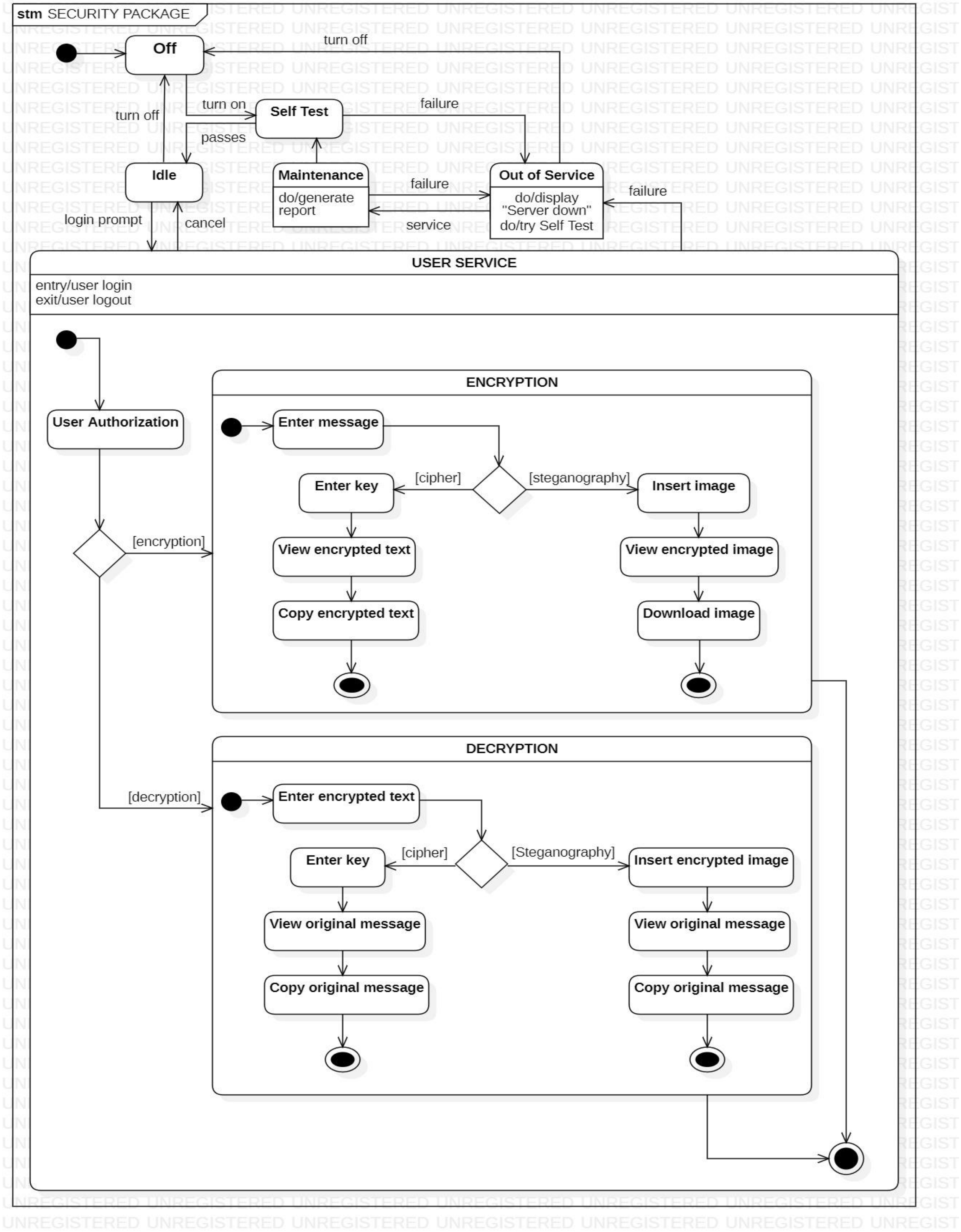
Portability:

- **Description** : the ability to be moved or reused in another environment
- **Solution :** Solution is to establish a platform independent system that can be incorporated into any application server

Security:

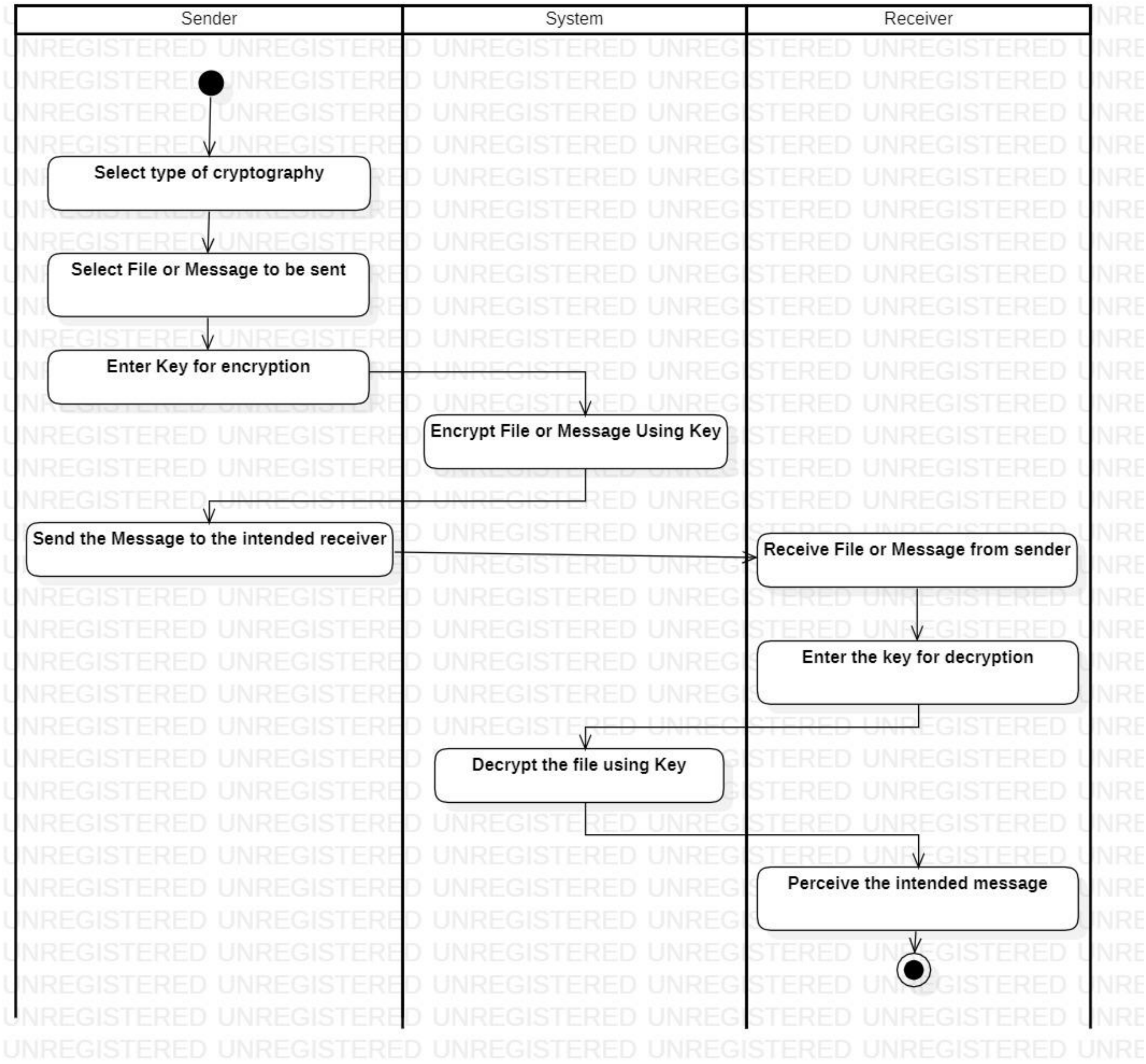
- **Description** : Authentication and authorization mechanisms
- **Solution :** Solution is to implement strong security standards

6 State Chart Diagram

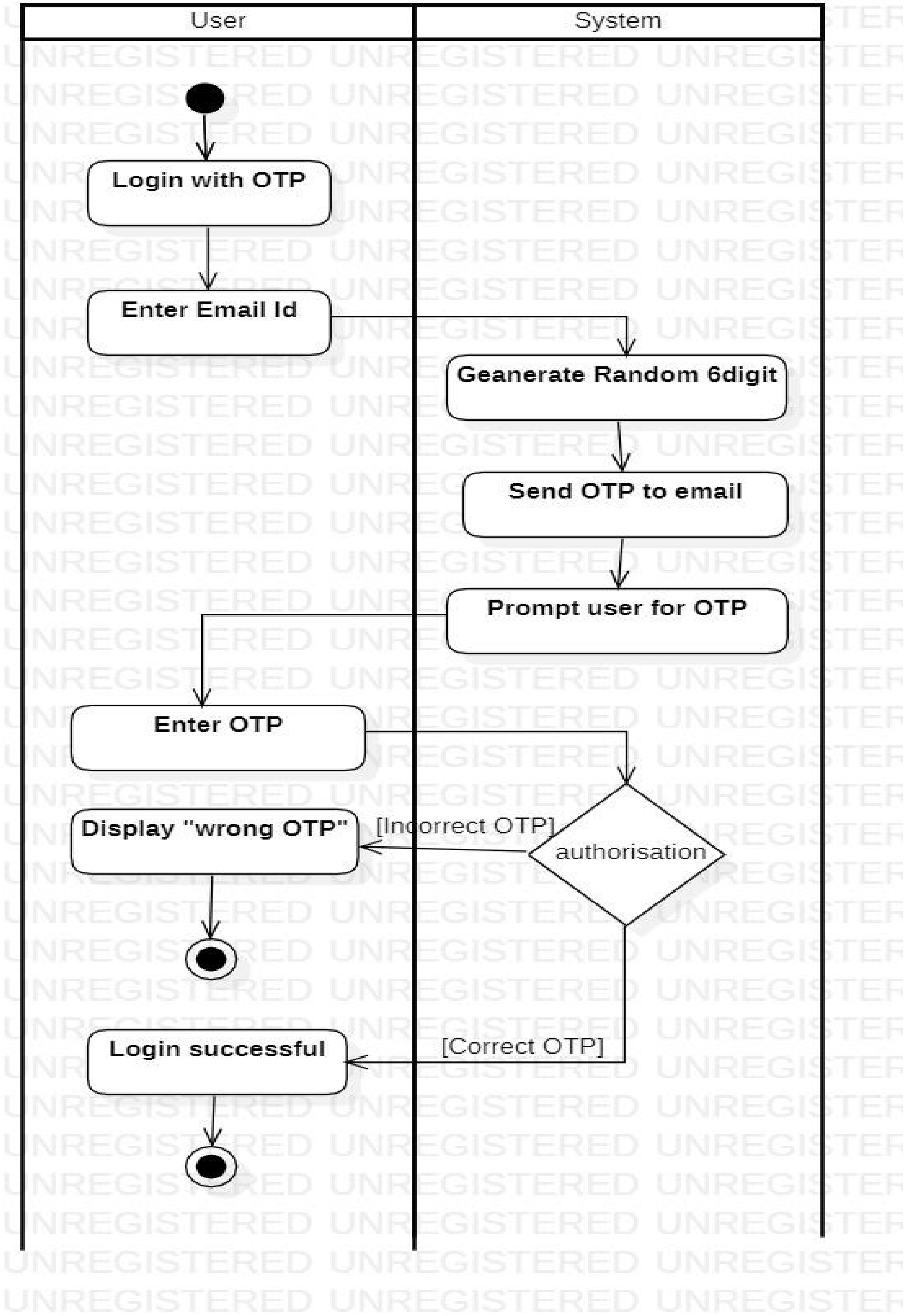


7 Activity Diagram

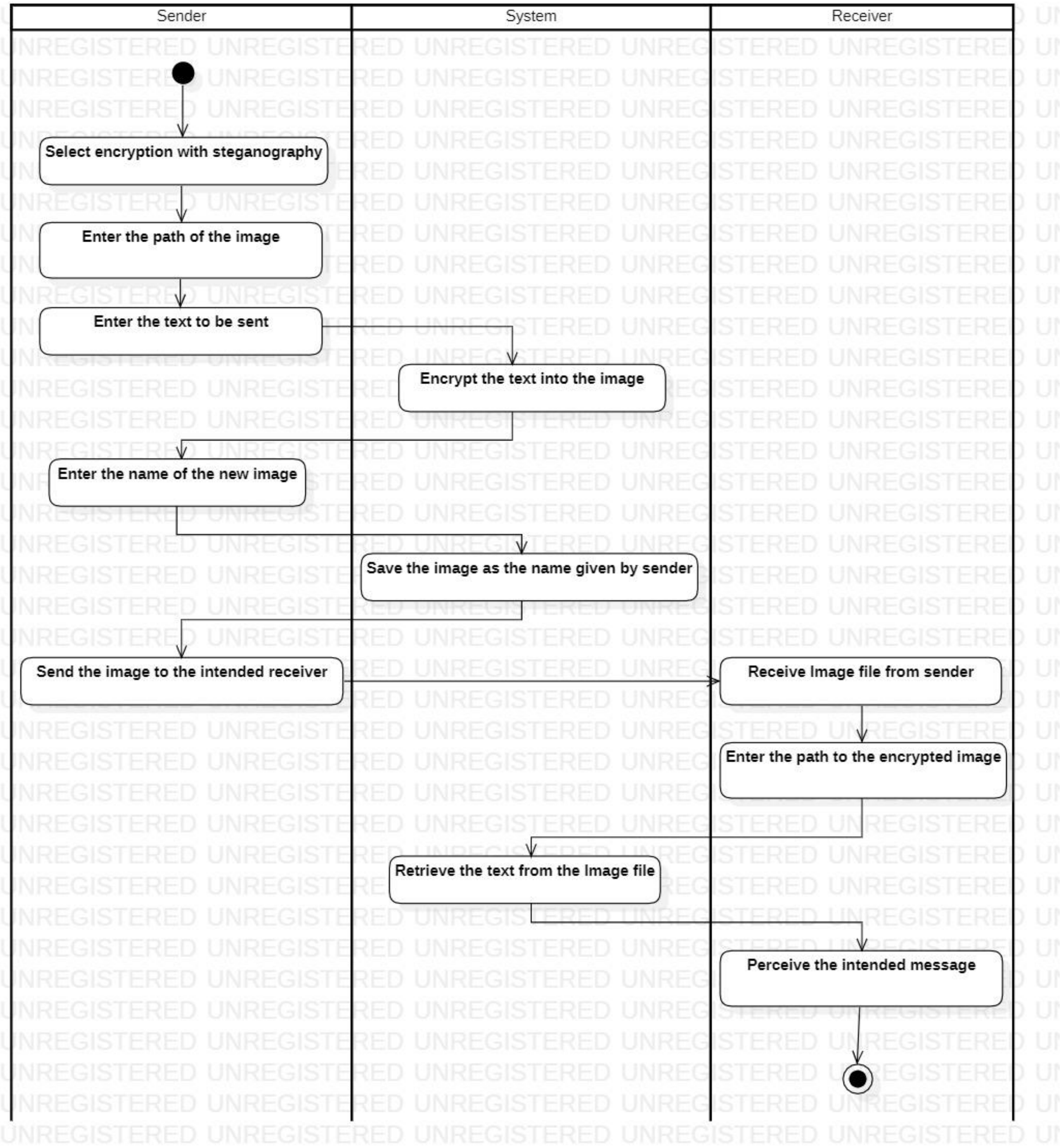
7.1 Activity Diagram for Encryption and Decryption


Sender
System
Receiver
Select type of cryptography
Select File or Message to be sent
Enter Key for encryption
Encrypt File or Message Using Key
Send the Message to the intended receiver
Receive File or Message from sender
Enter the key for decryption
Decrypt the file using Key
Perceive the intended message

7.2 Activity Diagram for OTP Generation and Authentication

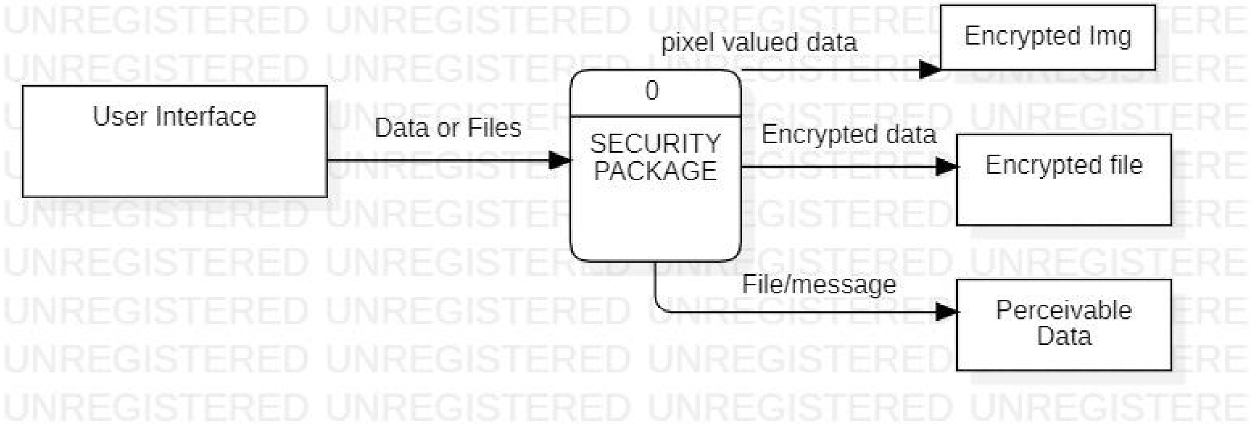


7.3 Activity Diagram to perform Steganography

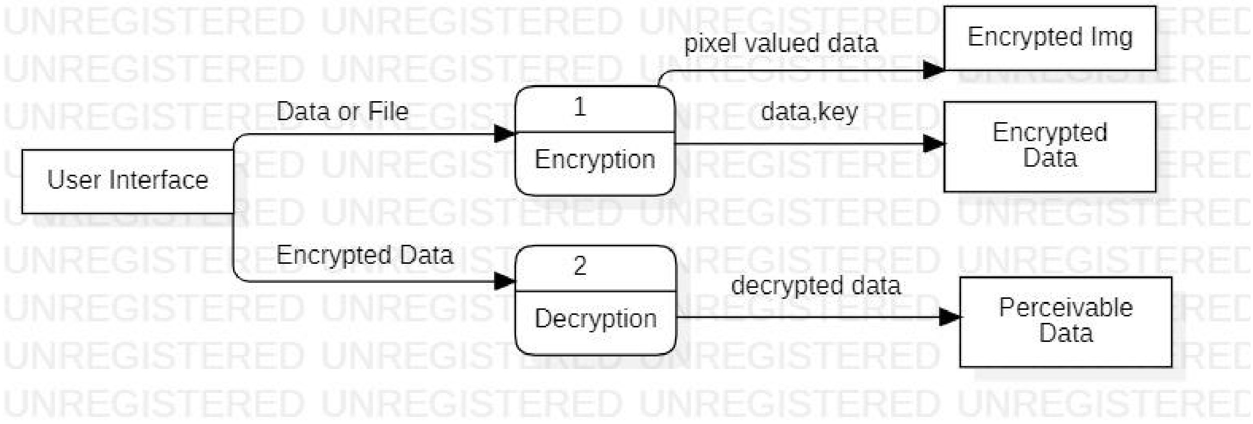

Sender
System
Receiver
Select encryption with steganography
Enter the path of the image
Enter the text to be sent
Encrypt the text into the image
Enter the name of the new image
Save the image as the name given by sender
Send the image to the intended receiver
Receive Image file from sender
Enter the path to the encrypted image
Retrieve the text from the Image file
Perceive the intended message

8 Data Flow Diagram

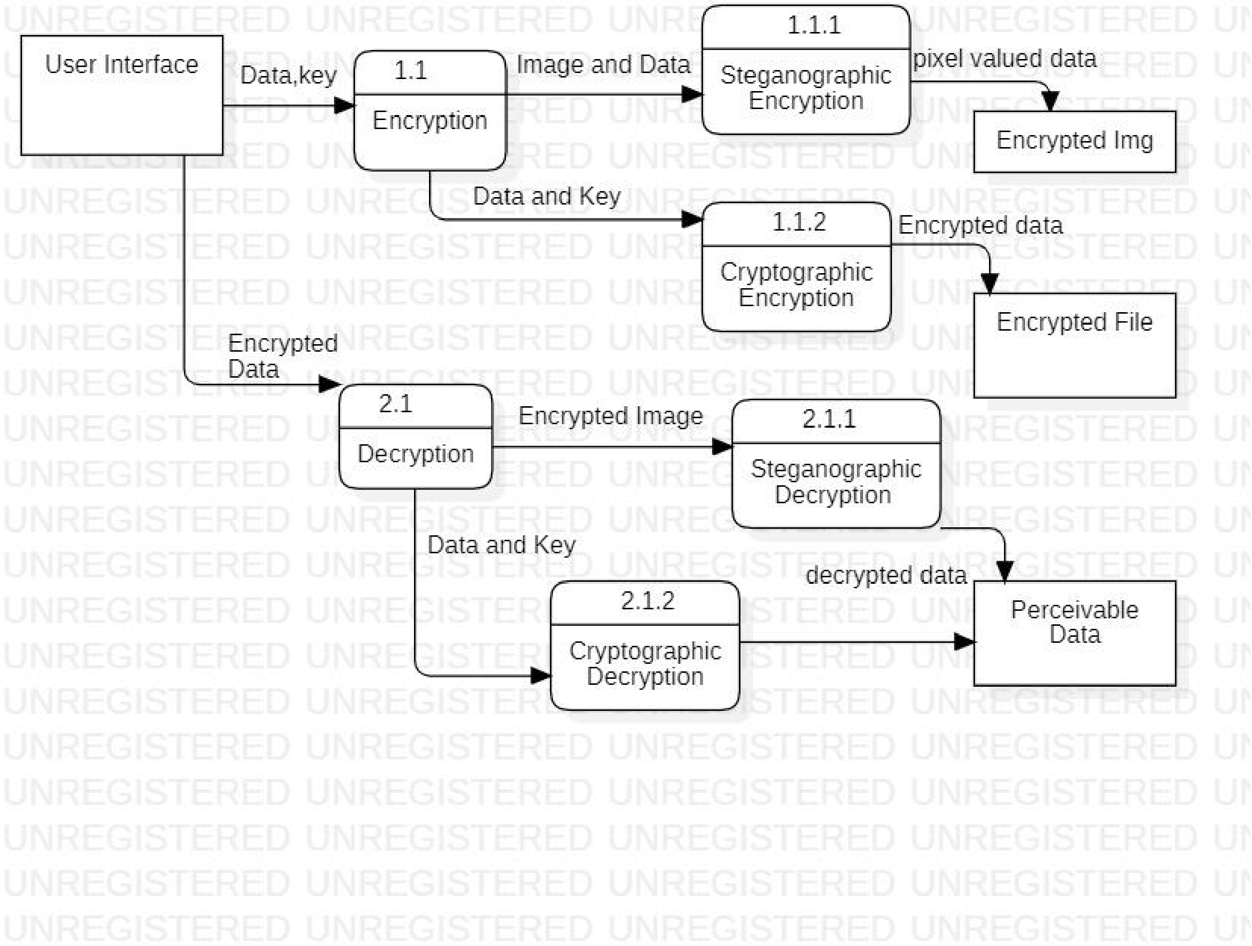
Level 0



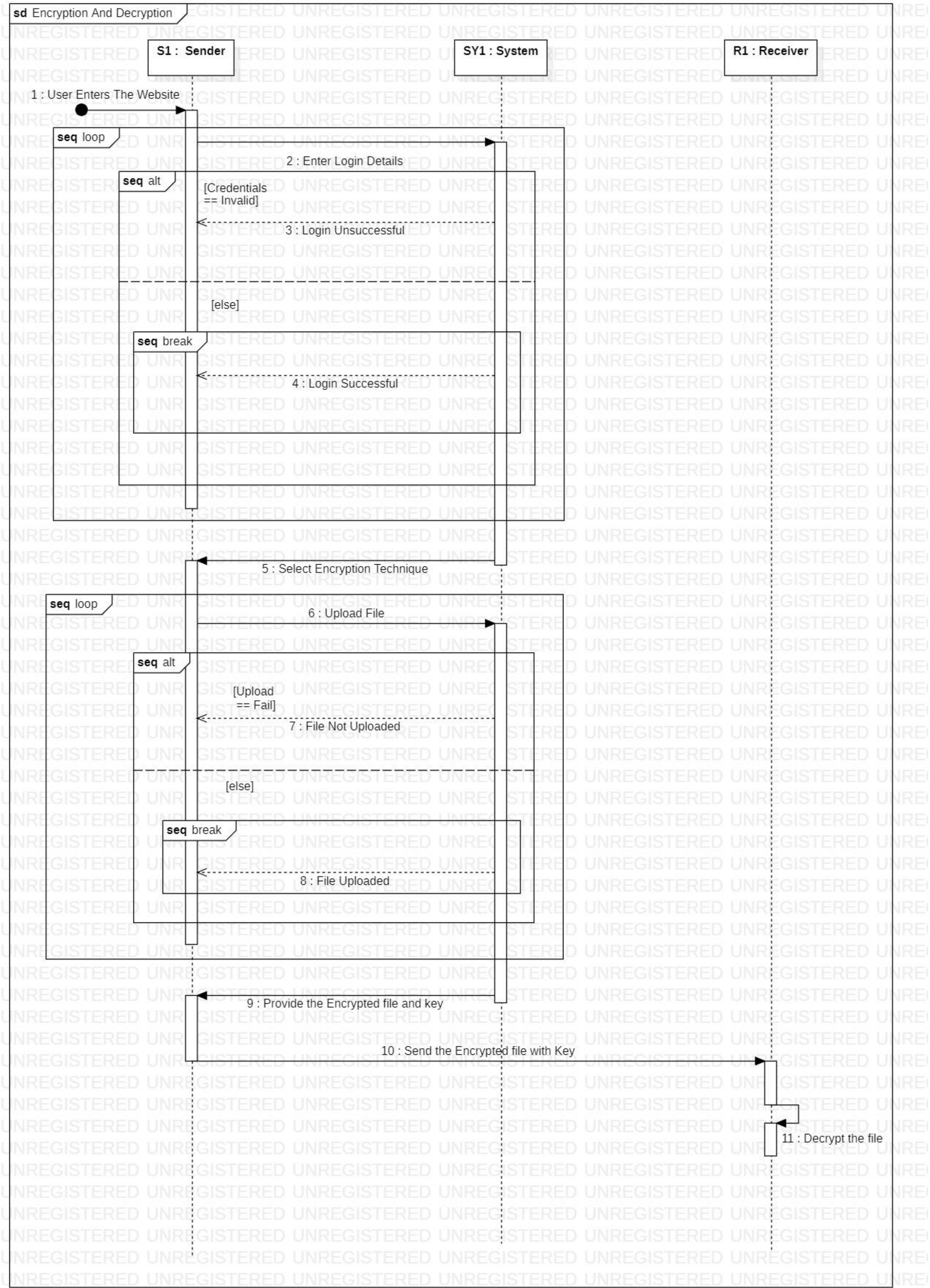
Level 1



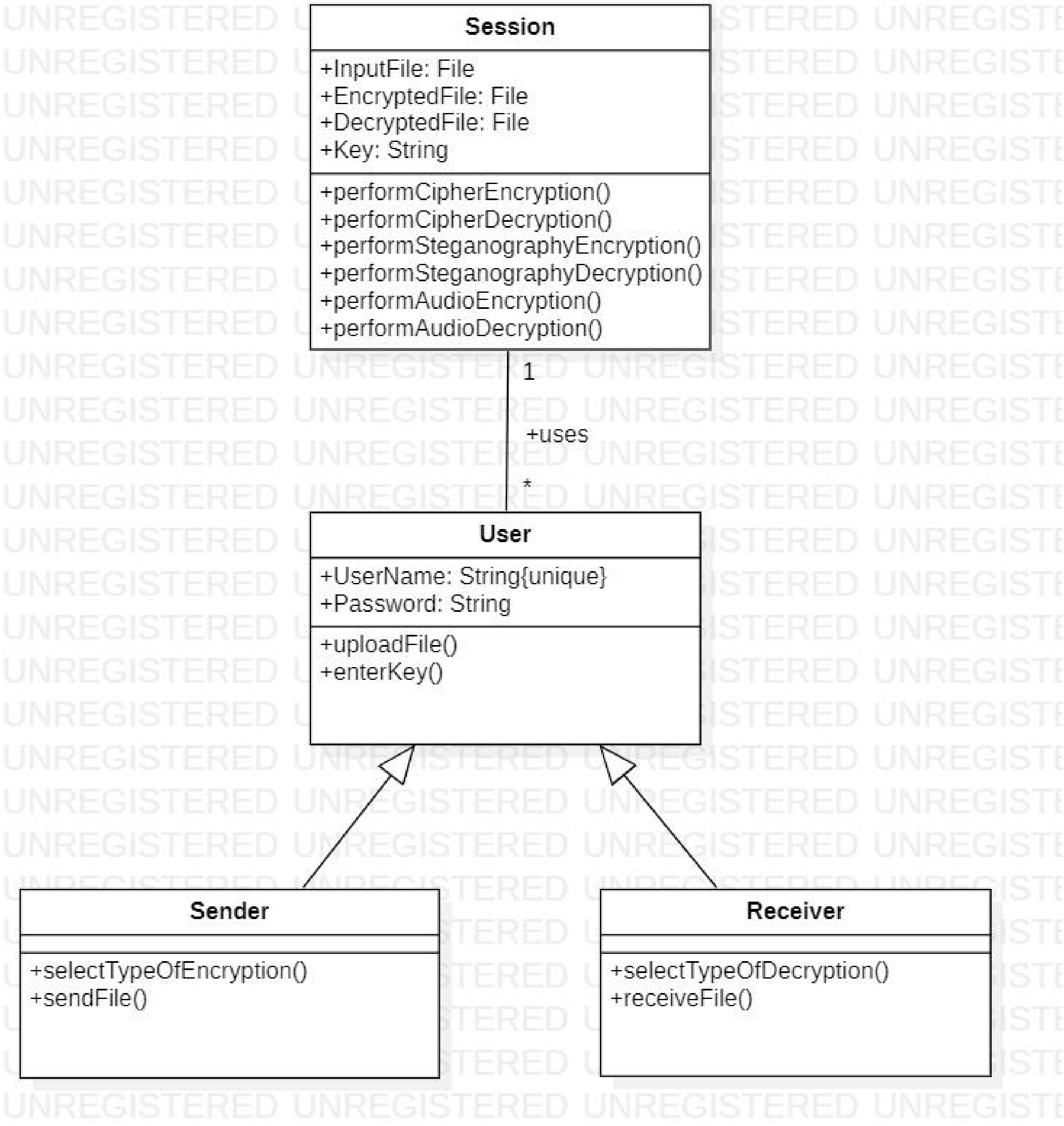
Level 2


User Interface
Data,key
1.1
Encryption
Image and Data
1.1.1
Steganographic Encryption
pixel valued data
Encrypted Img
Data and Key
1.1.2
Cryptographic Encryption
Encrypted data
Encrypted File
Encrypted Data
2.1
Decryption
Encrypted Image
2.1.1
Steganographic Decryption
Data and Key
decrypted data
2.1.2
Cryptographic Decryption
Perceivable Data

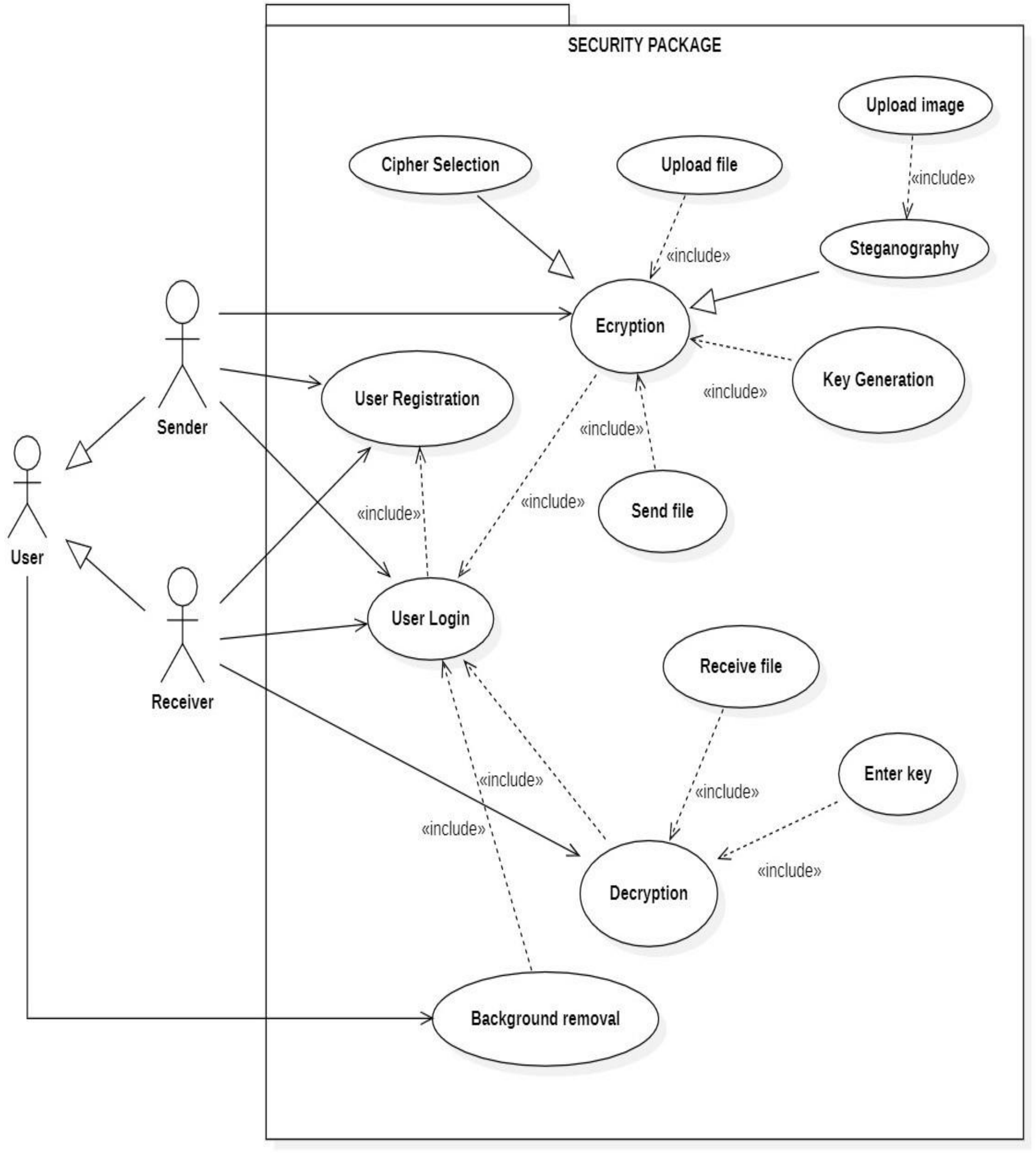
9 Sequence Diagram



10 Class Diagram


Session
+InputFile: File
+EncryptedFile: File
+DecryptedFile: File
+Key: String
+performCipherEncryption()
+performCipherDecryption()
+performSteganographyEncryption()
+performSteganographyDecryption()
+performAudioEncryption()
+performAudioDecryption()
1
+uses
*
User
+UserName: String{unique}
+Password: String
+uploadFile()
+enterKey()
Sender
+selectTypeOfEncryption()
+sendFile()
Receiver
+selectTypeOfDecryption()
+receiveFile()

11 Use Case Diagram



USE CASE DESCRIPTION TABLE

Use Case ID:	001		
Use Case Name:	Encryption		
Created By:	Team 7	Last Updated By:	Team 7
Date Created:	09-08-2022	Date Last Updated:	15-08-2022

Actor:	Sender
Description:	The sender encrypts the file and sends
Preconditions:	The sender must register and login
Postconditions:	The receiver receives an encrypted file
Priority:	High
Frequency of Use:	When the user needs to send an encrypted file to a receiver
Normal Course of Events:	1. Sender logins 2. Uploads the file to be sent 3. Select the type of cryptography 4. Generates key 5. Encrypts the file 6. Sends the file to the receiver
Alternative Courses:	Sender cannot upload the file
Exceptions:	File size cannot be too large.
Includes:	Upload file, Key Generation, Send File
Special Requirements:	The system must have sufficient internet connectivity to use the modules of the security package. The system should have basic hardware configurations. RAM: 4GB Operating system: Windows, Linux
Assumptions:	Sender sends the key to the receiver
Notes and Issues:	The sender can choose the type of cryptography from different ciphers. The sender generates a key every time a message is encrypted.

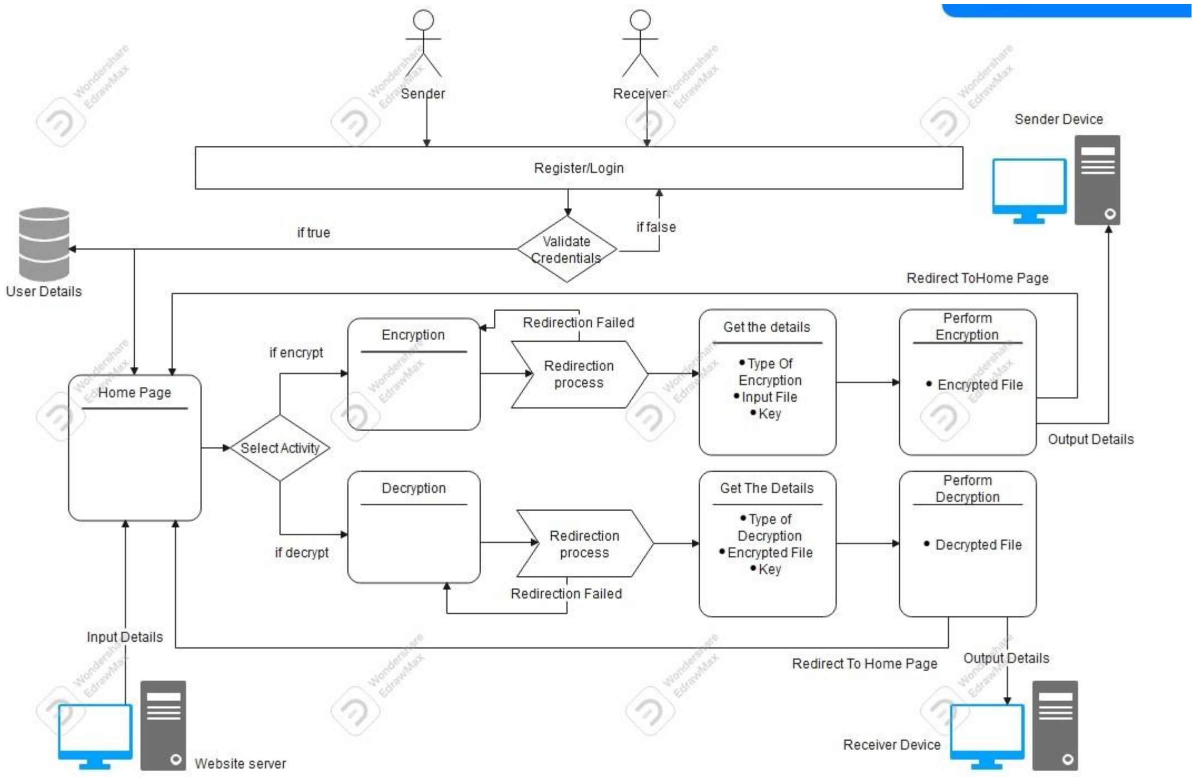
Use Case ID:	002		
Use Case Name:	Decryption		
Created By:	Team 7	Last Updated By:	Team 7
Date Created:	09-08-2022	Date Last Updated:	15-08-2022

Actor:	Receiver
Description:	The receiver receives the file and decrypts it
Preconditions:	The Receiver must register and login
Postconditions:	The receiver reads the decrypted message
Priority:	High
Frequency of Use:	When the user needs to read a decrypted message
Normal Course of Events:	1. Receiver logins 2. Receives the encrypted file 3. Enters the key 4. Decrypts the file 5. Read the message
Alternative Courses:	Receiver didn't receive the file and key
Exceptions:	Incorrect Key: Key doesn't decrypt properly
Includes:	Receive file,Enter Key
Special Requirements:	The system must have sufficient internet connectivity to use the modules of the security package. The system should have basic hardware configurations. RAM: 4GB Operating system: Windows, Linux
Assumptions:	Sender sends the key to the receiver
Notes and Issues:	The receiver gets the key from the sender and the receiver decrypts the file based on the encryption algorithm used.

Use Case ID:	003		
Use Case Name:	Steganography		
Created By:	Team 7	Last Updated By:	Team 7
Date Created:	09-08-2022	Date Last Updated:	15-08-2022

Actor:	Sender
Description:	Sender uses steganography to encrypt the file.
Preconditions:	The sender must register and login
Postconditions:	File encrypted using steganography
Priority:	High
Frequency of Use:	When steganographic encryption is necessary
Normal Course of Events:	1. Sender logins 2. Uploads the file to be sent 3. Uploads the image for steganography 4. Encrypts the file 5. Sends the file to the receiver
Alternative Courses:	Sender cannot upload the file or the image Sender cannot encrypt the file
Exceptions:	File size cannot be too large.
Includes:	Upload image
Special Requirements:	The system must have sufficient internet connectivity to use the modules of the security package. The system should have basic hardware configurations. RAM: 4GB Operating system: Windows, Linux
Assumptions:	Sender sends the key to the receiver
Notes and Issues:	Based on the key, the image can be decrypted on the receiver side.

12 Architecture Diagram


Sender
Receiver
Sender Device
Register/Login
if true
Validate Credentials
if false
User Details
Redirect ToHome Page
Redirection Failed
Encryption
if encrypt
Redirection process
Get the details
• Type Of Encryption
• Input File
• Key
Perform Encryption
• Encrypted File
Home Page
Select Activity
Output Details
Decryption
Redirection process
Get The Details
• Type of Decryption
• Encrypted File
• Key
Perform Decryption
• Decrypted File
if decrypt
Redirection Failed
Input Details
Redirect To Home Page
Output Details
Website server
Receiver Device