Bucharest University of Economic Studies
Faculty of Cybernetics, Statistics and Economic Informatics
IT&C Security Master Program

Steganography Mobile Application

Dissertation thesis

Scientific coordinator
Assoc. Prof. Popa Marius Emanuel Ph.D.

Master student
Ionescu Radu Ștefan

Bucharest 2020

Statement regarding the originality of the content

I hereby declare that the results presented in this paper are entirely the result of my own creation unless reference is made to the results of other authors. I confirm that any material used from other sources (magazines, books, articles, and Internet sites) is clearly referenced in the paper and is indicated in the bibliographic reference list.

Table of Contents

1. Introduction

For as long as we have known ourselves as humans, besides other problems such as food, water or shelter, we have also had the difficulty of communicating sensitive information. While this dilemma existed in our minds since the birth of humankind, rudimentary solutions started to be implemented back in the ancient times.

Nowadays, in an era of information technology, where virtually anybody owns some kind of device connected to the internet, which provides access to lots of data, digital steganography is the modern way of concealing messages or other delicate information. As opposed to ancient steganography techniques, which were implemented in a physical manner, using special ink, writing the message in an unobservable area of a letter or using certain rules for extracting the concealed information from an otherwise normal looking text, the modern implementations makes use of the digital form in which information circulates now.

The term steganography comes from the Latin word "steganographia" which originated from the Greek words "steganos", meaning concealed and "graphia", meaning writing, so steganography describes, generally, any kind of concealed writing.

Digital steganography is the process of embedding sensitive pieces of data into another computer file in such a manner that human perception cannot detect the difference between the original file and the one containing added information. For the purpose of this thesis, the confidential data will be referred to as the secret message and the container which embeds it will be called the cover file. Generally, both the secret message and the cover file can be of any computer file type. For example, we can hide: a plain text file into an image file, a picture into a sound recording or an image into a text file. However, it is not always possible to do this due to the sizes of the files, so hiding some text into an image file may be far easier than the other way around because, generally, an image file is bigger in size than a text file, but more important, a text file doesn't contain redundant information, in comparison with a picture. For this reason, this thesis will describe how to embed secret text messages in image and sound recording files. This computer mediums are generally large and redundant enough that we can alter them, by embedding the secret information inside, but to the human eye or ear it will appear the same. This is a clever way to send sensitive information because, while cryptography ensures the message is not readable, steganography does not reveal the existence of the message at all.

Additionally, encryption will be used in order to ensure better security for the message, by encrypting it before embedding. Cryptography is different than steganography because, while encrypting a message ensures its transformation into a form that an eavesdropper would not understand, steganography tries to hide the existence of a hidden message at all.

Other similar processes tightly coupled with steganography are fingerprinting and digital watermarking. A fingerprinting algorithm is used to generate a unique mark for a piece of data and embed it into that specific file. This is very useful when you want to supply some files and protect them from ongoing distribution. Watermarking also embeds a mark of the files with the purpose of signifying ownership. As opposed to steganography, in fingerprinting and watermarking the existence of the embedded data is publicly known, whereas steganography tries to completely hide that there is any information hidden inside. While a successful attack concerning a watermarking or fingerprinting algorithm consists of removing the watermark or fingerprint, basically removing the ownership protection, an attack on a steganographic system should detect and eventually extract the hidden data.

LSB or Least Significant Bit steganography is a technique which implies using the redundancy of the information as a place to store the data to be hidden. This is the reason multimedia files are so popular with steganography. They contain a fair amount of redundant information, whether we discuss about image, audio or video files, which makes the perfect home for secret messages. On the other hand, from a statistical point of view, if the message is embedded directly, as it is, into a cover file, it may be easily discovered, because altering the original file this way changes its statistical properties, if the secret message is large enough.

This is another good reason to also use cryptography when applying this kind of technique. Besides securing the content of the message, through encryption, the original message is transformed in a more random looking form, that of a ciphertext. This way, the statistical impact of the embedding process is drastically reduced because the randomness of the least significant bits is not altered heavily. Also, the ciphertext should be as evenly as possible distributed through the cover file, for a better security over statistical methods of examination.

2. General presentation

This thesis describes the development of an Android mobile application used for peer to peer communication, or more informal, chatting. The application uses Google Firebase as a backend service for multiple purposes like authentication, real-time database and persistent storage. The application should allow and facilitate users to create accounts and send and receive different kinds of messages.

The main information the application needs and uses is made up of the user details and the messages exchanged between them. This information is taken from different forms filled in by the users.

2.1. Mobile application

Users can communicate with each other using the mobile application, which should be up and running at any time. For a good user experience, the required data will be filled in by the users in simple, intuitive forms and the buttons and different options will be named as evocative as possible. The information presented to the users will be displayed in frames whose elements will be logically placed, for the ease of understanding.

A user must register to use the StegLock application by filling the registration form or by using a personal social account, like Facebook or Google. By using a social account, all the required data about the user will be taken from there, with his permission. If the client chooses to complete the registration form, he needs to enter his personal data, which is the full name, a nickname, an email address, a profile photo and to create a password. Filled in data is verified in real time and the form doesn’t allow the creation of accounts with flawed data, indicating which of the fields contain wrong information and what are the rules for each.

After registration, the client needs to log in to use the application, by entering the email and the password or by using his Facebook or Google social account. For a good user experience, the application allows persistent user authentication. This means that, unless the user specifically wants to log out, he will remain authenticated even if the mobile device is shut down or restarted. Once logged in, the main application menu is presented to the user. This menu allows four options. The options are chatting, displaying the user information, displaying application information and logging out.

To start chatting, after selecting the chat option from the main menu, the user is presented the chat frame, split into two tabs, the users tab and the chats tab. The users tab displays a list of all of the registered users and at the top a text box is used to search for users by their nickname. A tap on a user from the list will open a new conversation with that specific user. The chats tab displays a list of users with whom conversations already exist. For new users the chats tab will be empty until the users receives a message or sends one. A tap on a nickname from the chats tab will open the conversation.

The conversation frame displays the different kinds of messages sent between clients. The messages sent by the current user are displayed on the right side of the screen and the received messages are displayed on the left side. The application allows users to send text messages, photos, images or sound recordings.

In the bottom of the frame a text box is used for the text message and four buttons are used to either send the text message, choose an image, take a photo or record a sound. If the image button is pressed, the photo gallery is opened for the user to select the image he wants to send. After the image is selected, it is displayed in a new form and a text box allows the user to input a secret message to be embedded into the image, then the image can be sent by tapping the send button. Similarly, if the photo button is pressed, the process is the same, but instead of the gallery, the camera of the mobile device is opened for the user to take a photo. If the record a sound button is pressed, a new frame is displayed, containing two buttons, one for start/stop recording and one for playing the recorded sound. Also, like the image/photo frame, this one allows the user to input the secret message to be embedded into the recording.

The conversation frame displays text messages, images, photos and sound recordings. The sound recordings are displayed as play buttons and a press on a recording will begin the playback of the recording. A long press on either of images, photos or audio recordings will show a dialog displaying the secret message embedded into the file.

The user profile information frame, accessible from the main menu, displays all of the information about the client and allows him to edit his profile picture or nickname, by pressing the appropriate buttons. These buttons display a dialog in which the user should input the data to be changed or select an image, in case of profile photo editing. Also, the edited details are validated in real time and flawed information is indicated in the form along with the rules to be followed to fill in correctly.

The application information frame displays a description of the application and the signification of different symbols and buttons used in the application. Also, here can be found a short tutorial on how to use the chatting functionality and how are the secret messages being embedded into the cover files. The steganographic algorithms used to create images or sounds with embedded messages are shortly described here.

3. Informatic system requirements and analysis

3.1. System requirements

The functional requirements of the informatic system will be identified and modeled using use case diagrams, which outlines the way in which the system is used, by highlighting the actors and the actions that can be done by them.

Below, in *Figure 3.1*, the general use case diagram is shown along with a text description of each one, elaborating the purpose, the main actor, the preconditions, the basic and alternate flows and the usage frequency of the respective use case.

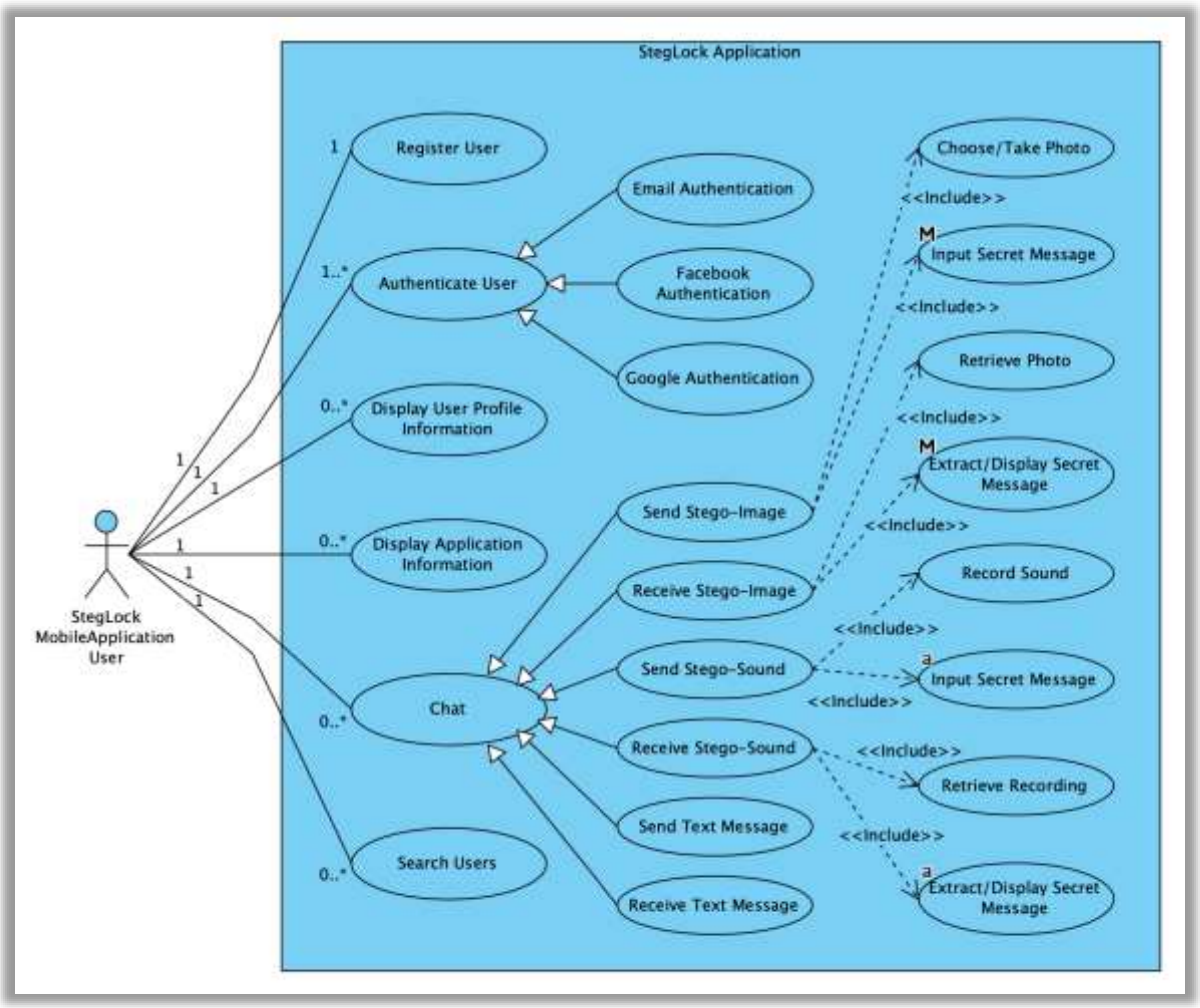


Figure 3.1 - Use case diagram

The above diagram covers all the possible usages of the system. The next chapter explains in detail the functionality of the system, by presenting each of the use cases present in the above diagram and detailing each one's specificities.

3.2. Use cases

The "Register User" use case has the purpose of creating an account for a user. The main actor, in this case, is the client, who starts this process. The completion of this use case is preconditioned by an internet connection and the client knowing his personal information needed to create the account. The basic flow of this use case is:

1. Choose profile picture
2. Input nickname
3. Input first name
4. Input last name
5. Input email address
6. Input password
7. Input confirmation password
8. Confirm data and create account

Alternatively, the user can cancel the process at any time and no account will be created. The usage frequency of this use case is low, because every client may have to complete this scenario once.

The "Authenticate User" use case has the purpose of signing in a client that has an account and it represents a generalization of the three main authentication methods. The main actor, in this case, is the application user, who initiates this scenario. The completion of this use case is preconditioned by an internet connection and the client knowing his credential information or social accounts credentials needed to sign into the account. The basic flow of this use case is:

1. Input email address
2. Input password
3. Authenticate

Alternatively, the client can use a social platform account, namely a Google or Facebook account, to get authenticated or he can cancel the process at any time and no account will be authenticated. The usage frequency of this use case is moderate to low, because every client may have to complete this scenario once or a few times as he will remain persistently authenticated until he wants to sign out.

The “Display User Profile Information” use case has the purpose of displaying account information to the client. The main actor is the application user, who initiates this scenario. The completion of this use case is preconditioned by an internet connection and the client being signed into his account. The basic flow of this use case is:

1. Select “My Profile” from menu
2. Display account details

The usage frequency of this use case is moderate to low, because every client may check his profile information from time to time, when changes about his information appear.

The “Chat” use case has the purpose of communication between the users and it represents a generalization of the multiple ways in which clients can exchange messages. The main actor, in this case, is the client, who starts this process. The completion of this use case is preconditioned by an internet connection and the client being authenticated. The specific chat use cases are “Send Text Message”, “Receive Text Message”, “Send Stego-Image”, “Receive Stego-Image”, “Send Stego-Sound” and “Receive Stego-Sound”.

The basic flow of the “Send Text Message” use case is:

1. Input text message
2. Send message

The basic flow of the “Receive Text Message” use case is:

1. Retrieve text message
2. Display message

The basic flow of the “Send Stego-Image” use case is:

1. Choose image or take photo
2. Input secret message
3. Embed secret message into image
4. Send image

The basic flow of the “Receive Stego-Image” use case is:

1. Retrieve image

2. Extract secret message from image
3. Display image
4. Display extracted secret message

The basic flow of the “Send Stego-Sound” use case is:

5. Record audio
6. Input secret message
7. Embed secret message into sound recording
8. Send recording

The basic flow of the “Receive Stego-Image” use case is:

5. Retrieve recording
6. Extract secret message from recording
7. Play sound recording
8. Display extracted secret message

Alternatively, the user can cancel the process at any time and no messages will be sent. The usage frequency of this use case is high, because this is the main use case of the system and client may chat frequently.

3.3. System analysis

The analysis of the system implies the examination of the requirements and use cases, both from a static and dynamic point of view, identifying and highlighting the basic concepts the system is working with and also, the existing relationships between them.

3.4. Static analysis

The static analysis of the system implies the development of the class diagram, by identifying the main classes, necessary for the operation of the system. In the analysis stage, a simplified class diagram is developed and later, in the system design stage the detailed class diagram will be developed, by specifying for each class the set of attributes, methods and constraints.

Analyzing the system’s use cases and requirements, two main classes are identified. Those are the “User” class and the “Message” class and in *Figure 3.2 – Simplified class diagram* those are presented, along with the identified relationships between them.

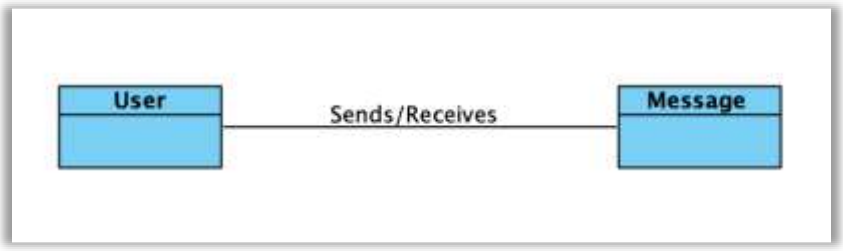


Figure 3.2 – Simplified class diagram

3.5. Dynamic analysis

The dynamic analysis of the informatic system implies observation of the evolution of its components over time, by developing some specific diagrams, namely state machine diagrams, activity diagrams and interaction diagrams.

The state machine diagram highlights the states through which an object or event can be found at certain moments in time and represents the lifecycle of the objects. The development of this kind of diagram also emphasizes the transitions between the states, by identifying the events that causes them.

For an object of User type, four possible states are identified, namely:

- Registered User
- Authenticated User
- Logged Out User
- Deleted User

In *Figure 3.3 – User state machine diagram* the states and transitions between User objects is detailed.

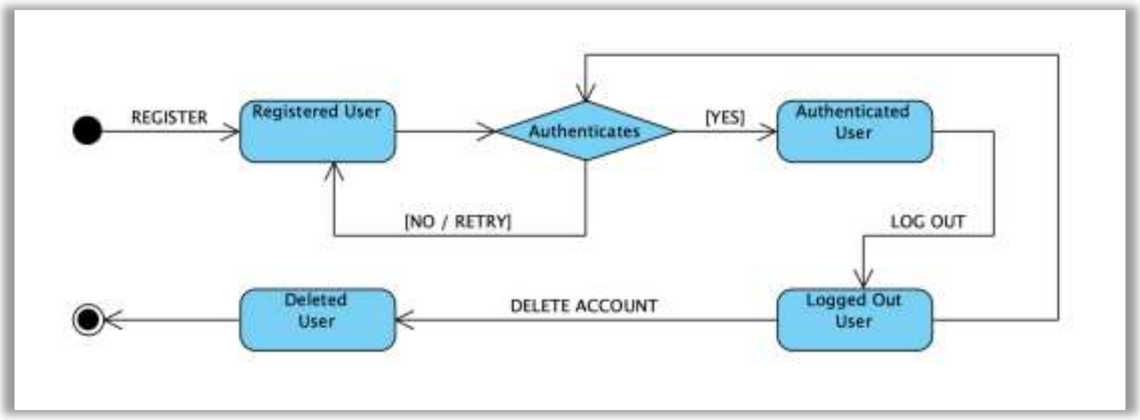


Figure 3.3 - *User state machine diagram*

For an object of Message type, six possible states are identified, namely:

- New Message
- Text Message
- Stego-Image Message
- Stego-Sound Message
- Sent Message
- Deleted Message

In *Figure 3.4 – Message state machine diagram* the states and transitions between User objects is detailed.

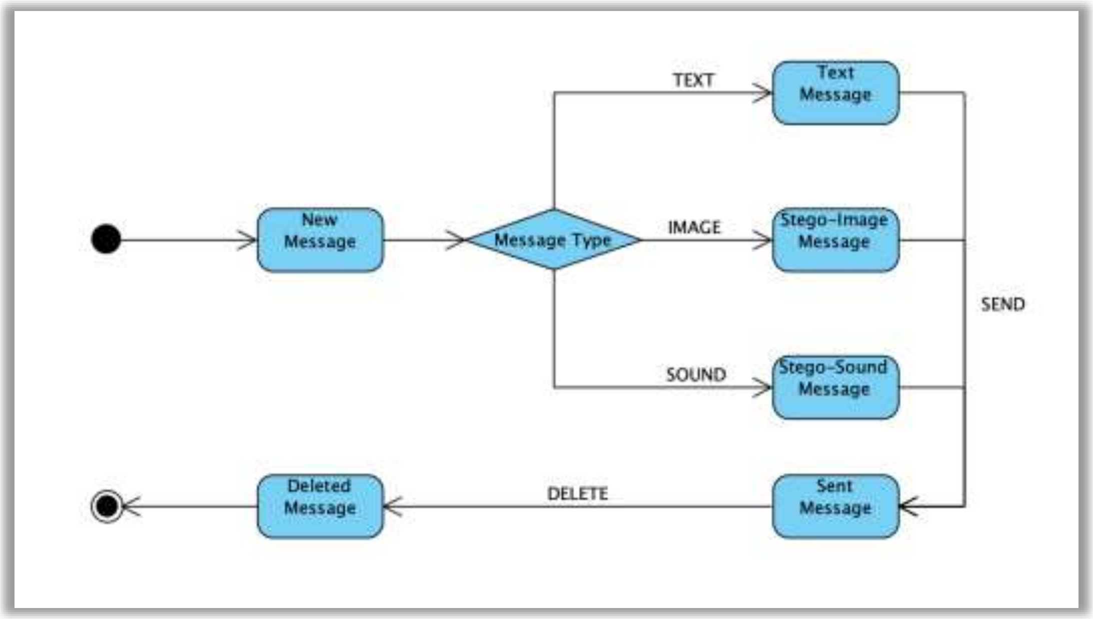


Figure 3.4 - *Message state machine diagram*

The activity diagram allows the highlighting of the work flows by identifying the action sequences and the results of those, also highlighting the decisions that may appear by executing the respective action.

Next, in *Figure 3.5 – Chat activity diagram*, it is presented the diagram of the chat activity, which is the main one. The diagram is made up of two vertical lanes, named partitions, representing the entities that participate in this activity. Also, there are multiple round boxes positioned inside the partitions, representing the actions to be made in this activity.

Control is passed from an action to another by using control flows or object flows. Control flows are represented by a simple arrow and object flows are represented by arrows along which square boxes are existing, boxes that represent the objects or data transferred between actions. The beginning and the end of the activity is marked by the initial and the final node.

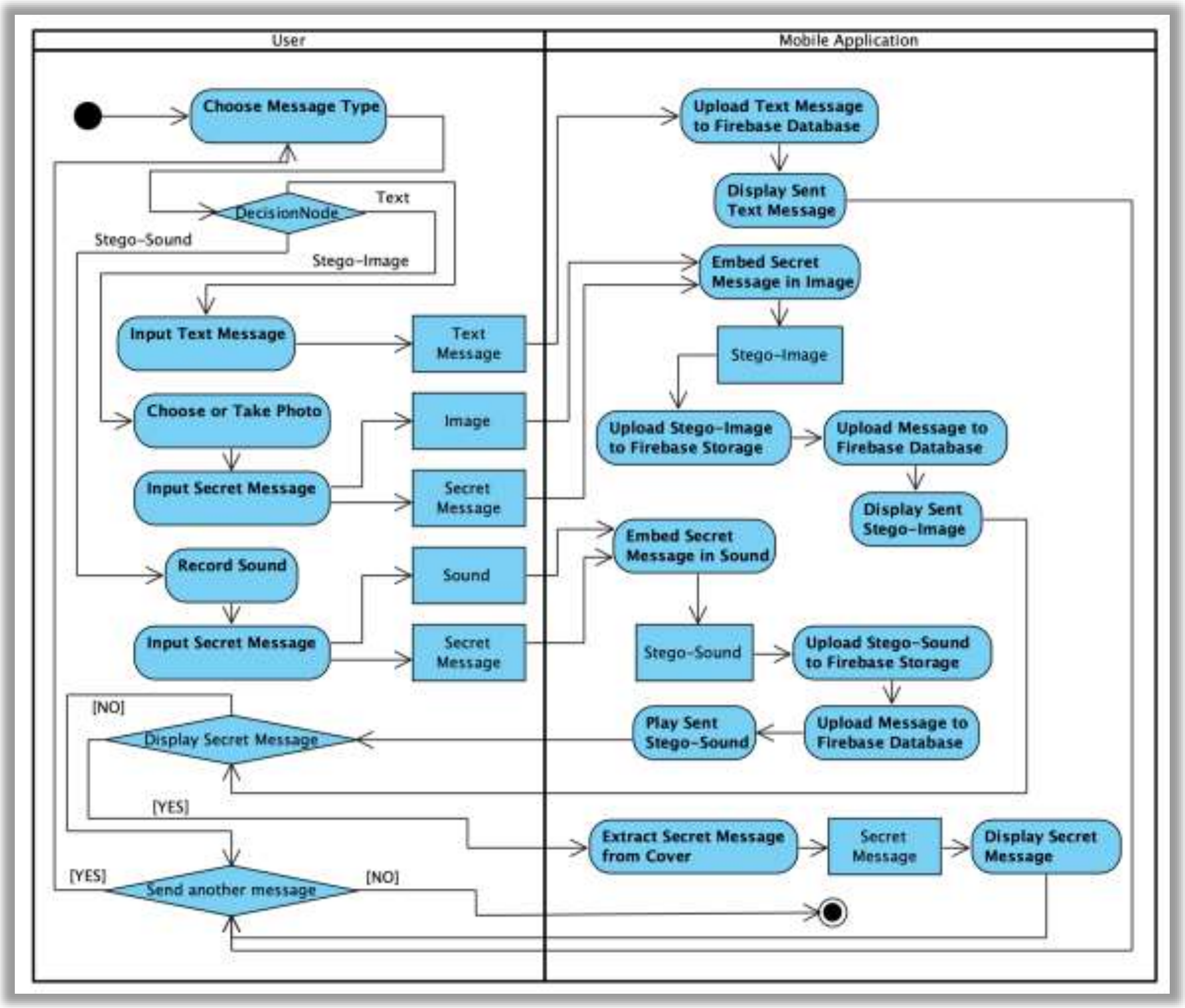


Figure 3.5 - Chat activity diagram

The interaction diagrams are highlighting the dynamic aspects of the system and are made up of a set of objects and the relationship between them, exposing in the same time their interaction, through the messages exchanged from one to another.

There are two types of interaction diagrams, the sequence diagram and the communication diagram. The two types of diagrams are equivalent from the point of view of their meaning, but both should be developed because the sequence diagram highlights the ordering and interaction between objects in time, while the communication diagram exposes the structure of the objects that are sending and receiving messages.

The sequence diagram is composed of the objects that interact with each other, placed on vertical lanes. Under these objects, their lifeline is represented using a discontinuous line, which signifies the existence of an object, in a certain period of time. Along the lifelines of the objects, tall rectangles mark an action done by the object, but also the start, end and duration of it on the timeline.

The messages sent between objects are represented by horizontal arrows which are crossing the vertical lifelines. Those arrows are connecting two control points of the various objects which communicate or they can point to the same object, signifying self-communication. At the same time, those arrows also indicate the moment in time at which the action is done, through the place they are placed in.

Next, in *Figure 3.6 – Chat sequence diagram*, the diagram is shown, detailing the objects that are participating in the chat action along with the communication that is taking place between them and their evolution in time.

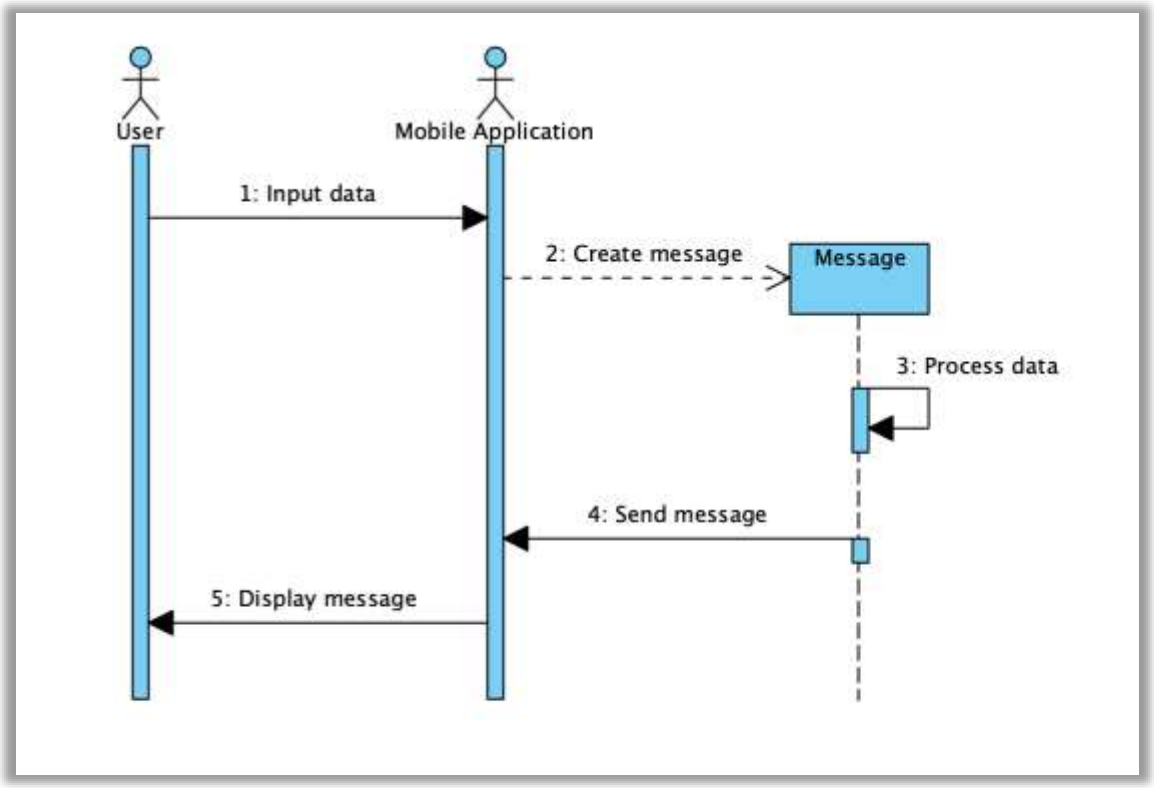


Figure 2.6 - Chat sequence diagram

The communication diagram emphasizes the structure of the objects and actors which are sending and receiving several messages. This diagram is represented like a graph whose nodes represent the participants and whose edges represent the connections between them. Along the edges between objects, there are the sent messages, prefixed by an order number. The direction of the messages is highlighted by using arrows.

In *Figure 3.7 – Chat communication diagram*, the diagram is presented, highlighting the messages exchanged.

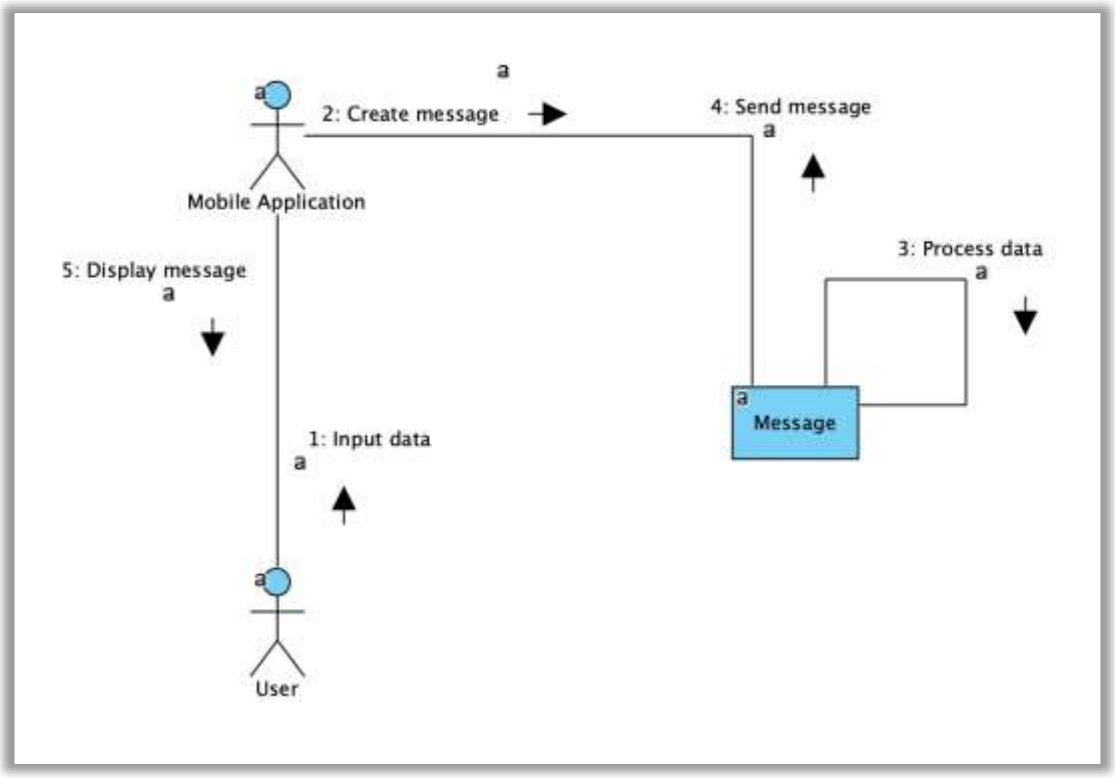


Figure 3.7 - Chat communication diagram

4. Informatic system design

4.1. Detailed class diagram

The detailed class diagram highlights the main classes which make up the informatic system, along with the attributes and methods that are characterizing them. Both the attributes and the methods are described by their names, but usually, the diagram shows many other characteristics of them, like visibility, type, multiplicity, default value, parameter list or return type. Furthermore, the diagram presents the relationship between those classes, which can be unary, binary or ternary, based on the number of them which are interacting. A binary relationship can express association, aggregation, generalization or dependency.

In *Figure 4.1 – Detailed class diagram*, the diagram of the classes is presented, along with the relationship between them and the name and multiplicity of those.

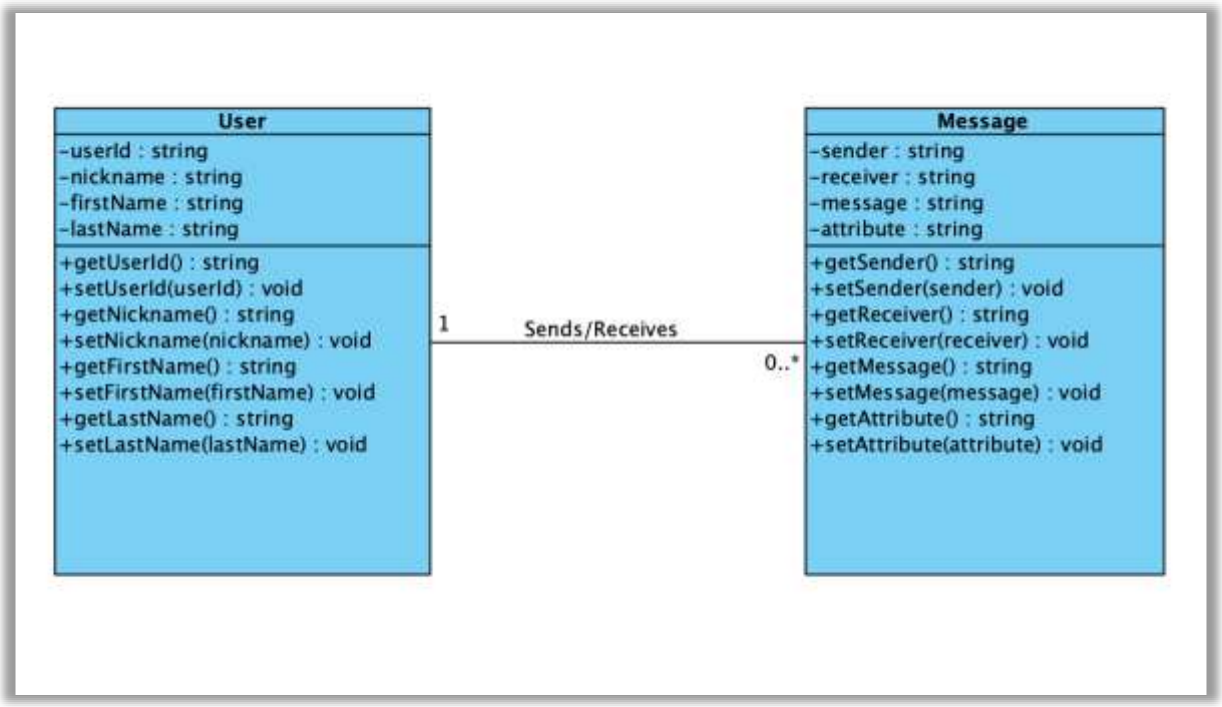


Figure 4.1 - Detailed class diagram

4.2. Database design

Following the analysis of the functional requirements of the system, two entities that compose the data model were identified, those being the user and the message. Those entities were described in the previous chapter.

For persistent storage the system will use a third-party database service, namely Firebase Realtime Database from Google. This database is a fast NoSQL cloud-hosted database and the data is persistently stored in JSON (Java Script Object Notation) objects and synchronized in real time to all the clients. This database is also optimized for offline use, so if a client loses his internet connection the data will be cached and synchronized when the connection is established again.

For keeping the database secure, some security rules can be defined to specify which clients have access to the data and more specifically to what data are those clients having access. The database is stored in the cloud so there is no server implementation or operations required.

The basic setup of the database is done in the Firebase web console, along with the security rules. The creation and manipulation of the user and message entities are realized inside the mobile application, using Google's Firebase software development kit, which include various libraries like Firebase Realtime Database, Firebase Storage and Firebase Authentication. The entities are created when needed, and updated or deleted accordingly using an Database Reference object, supplied by the Firebase Realtime Database library.

As described in the detailed class diagram, the user entity should have a unique user id, a first name, a last name and a nickname. The unique ids of the users are managed by the Firebase Authentication library and the rest of the data is collected from the user, at registration time, or from the social platform account used to authenticate.

The message entity should have a sender, a receiver and a message. The sender and receiver attributes of the message are the unique ids of the users which are communicating. If the message sent between the users is a plain text message, the message attribute will contain the sent message. If the message is consisting of a steganographic image, photo, or sound recording, the message attribute will contain the Firebase Storage relative path, which indicates where to find the sent multimedia object.

When choosing to send an image, photo or sound recording, those will be uploaded to the Firebase Storage, using the specific library. The sent multimedia files are manipulated, uploaded or deleted using a Storage Reference object from Google's Firebase Storage library.

Before uploading the multimedia object to storage, inside the mobile application, the user's secret message is embedded into the image, photo or sound recording, to be later extracted by the receiver of the message, if wanted.

4.3. User interface design

For a good user experience and a usage of the application as simple as possible, the graphic interface should be rigorously designed and all the buttons, options, menus and display components should be simple and intuitive. The user interface elements should operate as described in this chapter.

When running the application, the client is presented the log in activity. This activity is made up of an input form for authentication and button control for registration. The authentication form consists of two text input controls, one for the email address and one for the password, a button control for submitting the input data and two button controls used for Facebook or Google authentication.

If the user is new and does not have an account, by pressing the registration button control, the registration form will be displayed. On this form there are 6 text input controls which allow the user to input the data needed for the account to be created. Also, at the top of the form an image control permits choosing a profile picture. The data inputted in the controls is validated in real time and if it is not correct the wrong data is highlighted, along with the rules and the user has to correct it before the registration can be completed. After the registration is successful, the user is presented again the log in activity, where he can use his credentials to log in.

After the user authenticates, the main menu of the application is presented. This menu is composed of four button controls, for chatting, profile information display, application information display and logout.

The user can logout only by pressing the specific button, otherwise remaining authenticated persistently, even if the application or mobile device is shut down or restarted.

The user profile information button control presents to the user the profile activity. This activity displays the information about the user account in text display controls. Also, this activity presents two button controls, which the client can use to change the editable components of the profile information, those being the nickname or the profile picture.

The application information activity displays data about the application. This data refers to how the application should be used and how the secret message is embedded into the cover file, using the steganographic algorithms.

The chatting button control displays the chat activity. This activity displays two tabs, one for the chats and one for the users. The users tab presents a list of all the users of the application, along with the profile picture of each. At the top of the tab, above the list, a text input control is displayed, where clients can input nicknames to search for a specific user in the list. A tap on a user in the list will open a new conversation with that user, unless one already exists. If a conversation with that user is already done, it will be opened. The chats tab displays all the current conversations the user has, as the nickname and profile picture of the client with whom he conversates. For a new user, the chats tab will be empty until the users sends a message to someone or receives one. A click on the nickname in the chats tab will open that specific conversation.

The conversations between users are displayed into the message activity. The message activity consists of a list view control, which displays in rounded boxes the different types of messages that can be sent. The messages sent by the current user are displayed on the right side and the received messages are displayed on the left side. This activity presents, under the list, a text input control where the user can input a text message to be sent and four button controls. The first button, next to the text input control is used to send the inputted plain text message. The second, third and last buttons are used to select an image from the gallery of the device, take a new photo or record a sound. The chosen action will provide a multimedia object that will be later used in the steganographic process to embed the secret message into it.

After an image is chosen from the gallery, or a new photo is taken the steganographic image activity is displayed. In this activity the user can input in a text field the secret message to be embedded into the image. Also, a button control is presented at the bottom of the activity, which if pressed begins the steganographic process and sends the image after its done. Similarly, if the recording button control is pressed, the steganographic sound recording activity is displayed, where the user can record and send an audio file with a secret message embedded.

4.4. Component diagram

The component diagram presents the modules of which the informatic system is composed. These components may be represented by source code files, binary files, function libraries, executable files and other types of files.

The presented system is composed mainly of Java files, which are containing the necessary source code, and XML (Extensible Markup Language) files describing the layout of the user interface, but also some libraries from Google like Firebase Authentication, Storage and Database and some other libraries from third parties. In *Figure 4.2 – Component diagram* those modules are presented along with the connections between them.

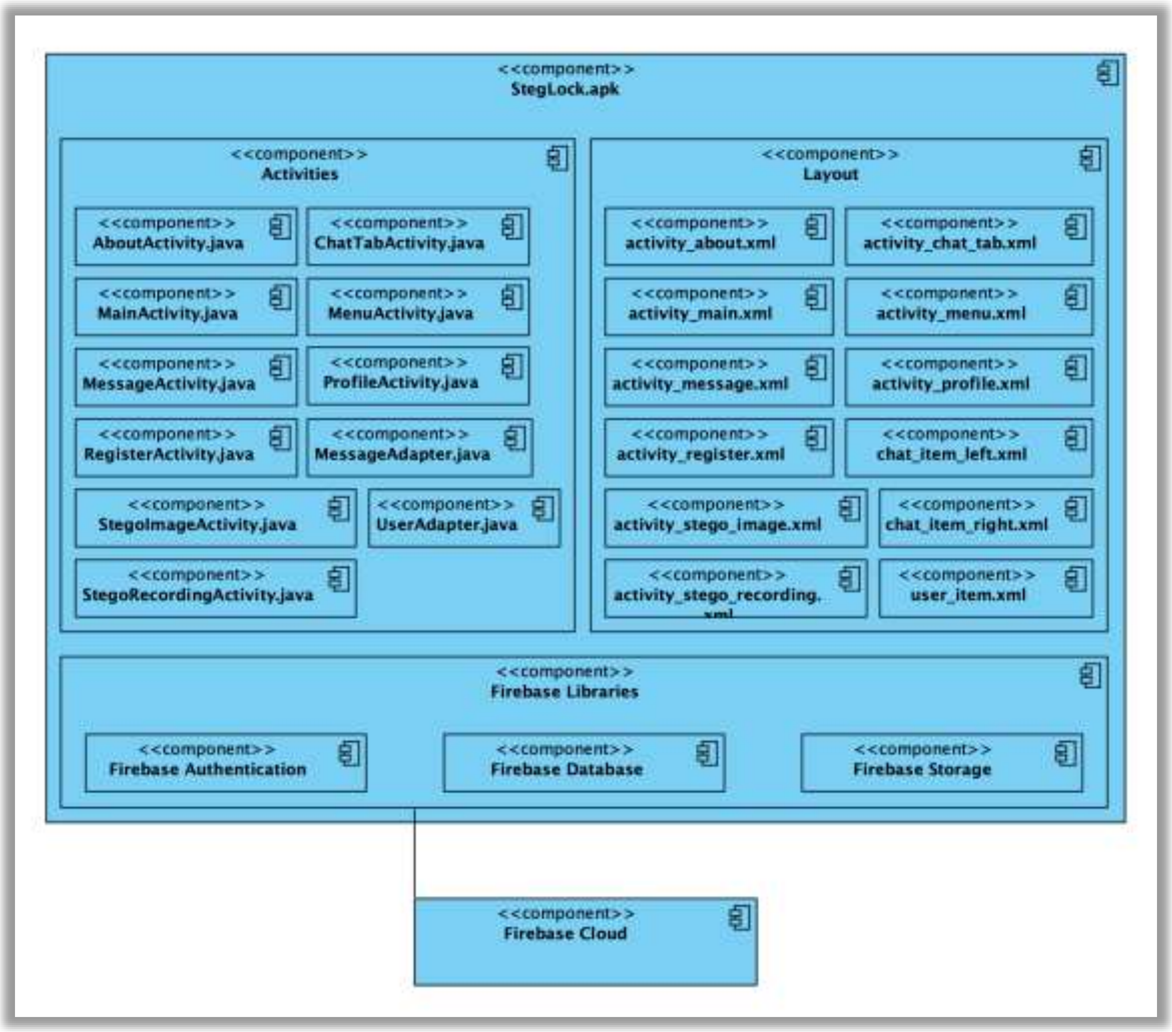

<<component>> StegLock.apk
<<component>> Activities
<<component>> AboutActivity.java
<<component>> ChatTabActivity.java
<<component>> MainActivity.java
<<component>> MenuActivity.java
<<component>> MessageActivity.java
<<component>> ProfileActivity.java
<<component>> RegisterActivity.java
<<component>> MessageAdapter.java
<<component>> StegoImageActivity.java
<<component>> UserAdapter.java
<<component>> StegoRecordingActivity.java
<<component>> Layout
<<component>> activity_about.xml
<<component>> activity_chat_tab.xml
<<component>> activity_main.xml
<<component>> activity_menu.xml
<<component>> activity_message.xml
<<component>> activity_profile.xml
<<component>> activity_register.xml
<<component>> chat_item_left.xml
<<component>> activity_stego_image.xml
<<component>> chat_item_right.xml
<<component>> activity_stego_recording.xml
<<component>> user_item.xml
<<component>> Firebase Libraries
<<component>> Firebase Authentication
<<component>> Firebase Database
<<component>> Firebase Storage
<<component>> Firebase Cloud

Figure 4.2 - Component diagram

4.5. Deployment diagram

The deployment diagram exposes the structure of the system at the execution moment, by highlighting the various equipment necessary for the operation of the system, along with the connections between them.

The systems necessary components are:

- Mobile Device – the mobile device of the client, on which the StegLock mobile application will run.
- Firebase Development Platform – the development platform from Google which provides necessary services like Authentication, Storage and Database.
- Internet Network – the network which connects all of the equipment.

In *Figure 4.3 – Deployment diagram* the system's necessary equipment is presented.

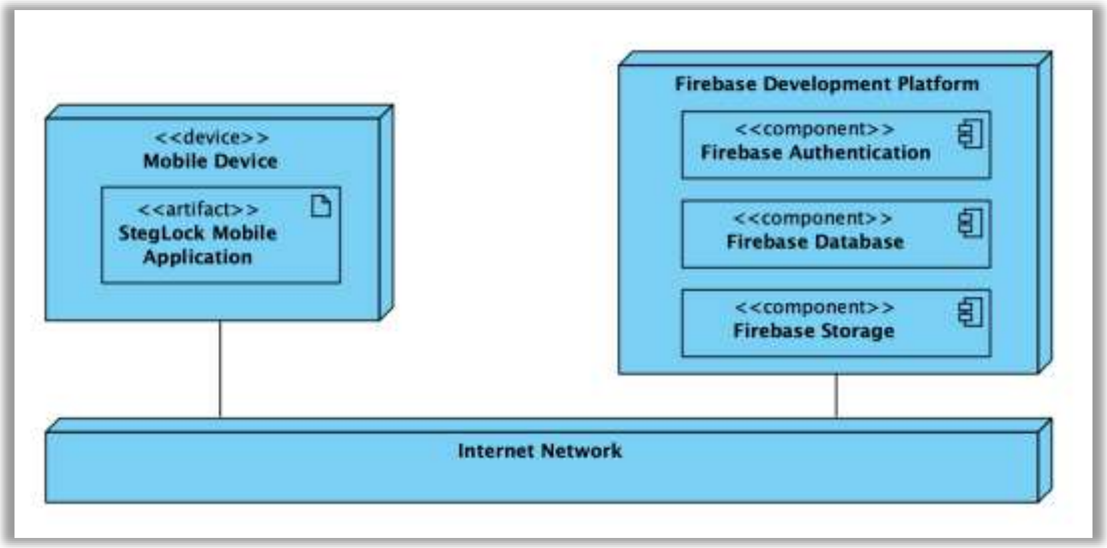


Figure 4.3 - Deployment diagram

Following the system design and analysis of the system requirements the implementation will be done accordingly, as described in all the diagrams presented in the last two chapters. This will be done using an appropriate, high-level programming language, namely Java, suitable for the Android execution environment.

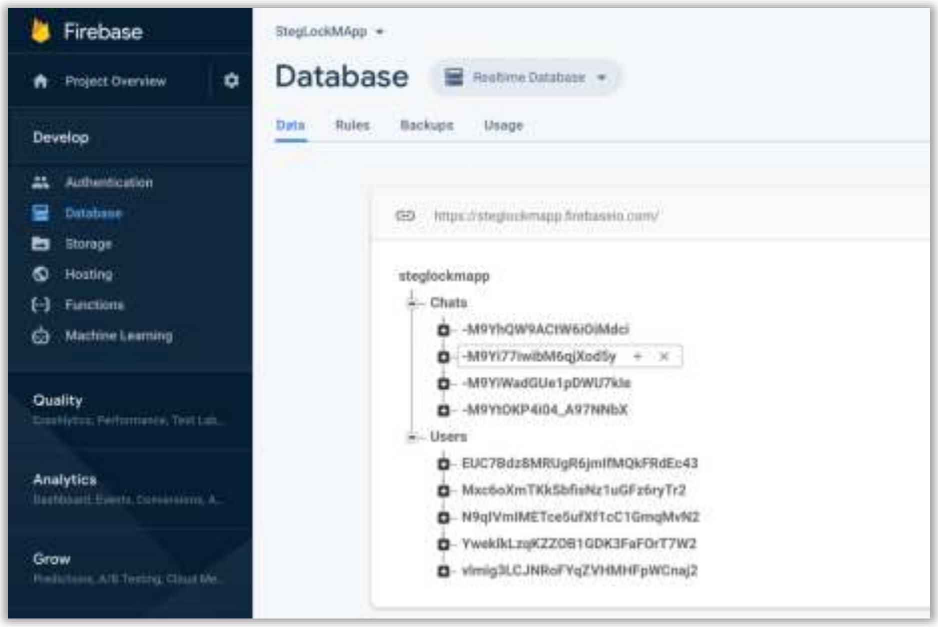
5. Application implementation

The developed system is implemented using the Java object oriented, platform independent programming language. The management of the user accounts is done using Google's Firebase Authentication libraries. Also, all the multimedia files the application is manipulating are saved in Firebase Storage, using the specific library.

The application uses Google's Firebase Database to persistently store data about the users and the messages, which a fast NoSQL database that stores the information as JSON files. The setup of the project is done in the Firebase web console.

The design of the informatic system was realized using a computer aided software engineering tool named Visual Paradigm. This software tool was used for the development of all the UML diagrams necessary for the creation and understanding of the system. The tool permits the in-depth design of the components of the system, specifying all the details of the system.

For the development of the Android mobile application, the Android Studio integrated software development environment was used. For the simplicity of development and lack of database connectivity in the Android native Java libraries, a third-party database service was used, namely Firebase Database, which can be set up very fast and straightforward using the Firebase web console. In *Figure 5.1* the Firebase Database web console is presented.



```
steglockmapp
 ├─ Chats
 │   ├─ -M9YhQW9ACtW6iOiMdci
 │   ├─ -M9Yi77iwibM6qjXod5y
 │   ├─ -M9YiWadGUe1pDWU7kle
 │   └─ -M9YtOKP4i04_A97NNbX
 └─ Users
     ├─ EUC7Bdz8MRUgR6jmIfMQkFRdEc43
     ├─ Mxc6oXmTKkSbfisNz1uGFz6ryTr2
     ├─ N9qlVmIMETce5ufXf1cC1GmgMvN2
     ├─ YwekIkLzqKZZ0B1GDK3FaFOrT7W2
     └─ vlmig3LCJNRoFYqZVHMHFpWCnaj2
```

Figure 5.1 - Firebase Database web console

5.1. Steganographic algorithms

Digital steganography is the practice of using some digital formats, such as image files or sound files, as a cover for hiding messages, without altering the original files in terms of human perception. This method of concealing data may use any computer file type as a container in which we can embed the same or another digital file type. For example, we can hide: a plain text file into an image file, a picture into a sound recording or an image into a text file. However, it is not always possible to do this due to the sizes of the files, so hiding some text into an image file may be far easier than the other way around because, generally, an image file is bigger in size than a text file, but more important, a text file doesn't contain redundant information, in comparison with a picture.

The mobile application embeds secret text messages in image and audio files. These digital mediums are generally large and redundant enough that we can alter them, by embedding the secret information inside, but to the human eye they will appear the same. This is a clever way to send sensitive information because, while cryptography ensures the message is not readable, steganography does not reveal the existence of the message at all. For a better security, cryptography will also be used to encrypt the messages before embedding them into the cover. In *Figure 5.2* the steganographic schema is presented.

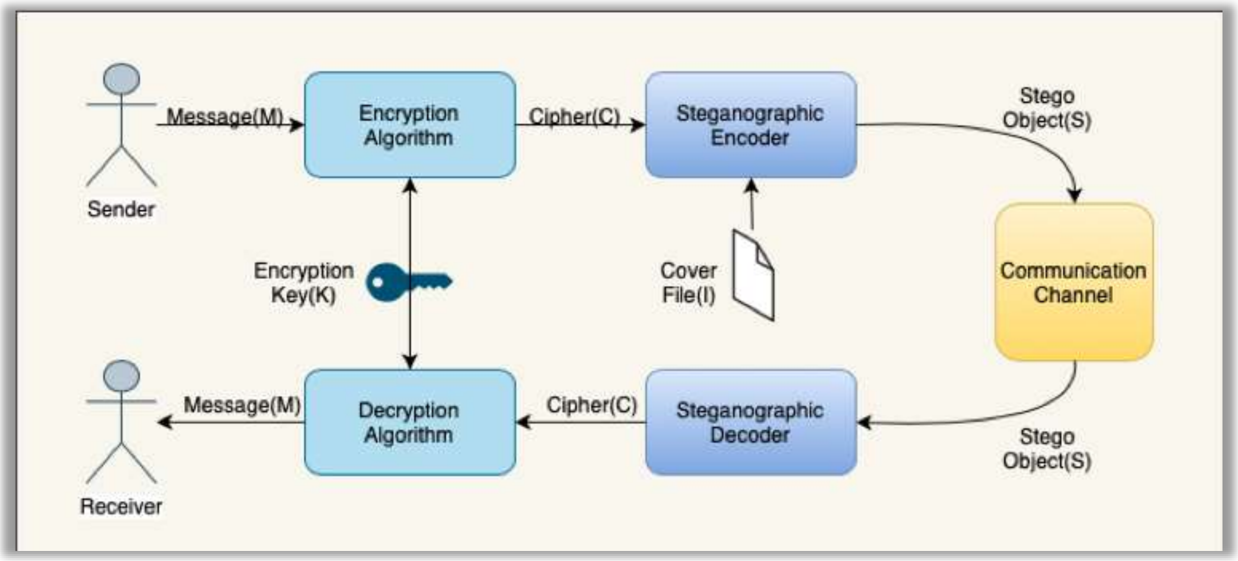


Figure 5.2 - Steganographic schema

From a statistical point of view, if the message is embedded directly, as it is, into a cover file, it may be easily discovered, because altering the original file this way changes its statistical properties, if the secret message is large enough.

This is another good reason to also use cryptography when applying this kind of technique. Besides securing the content of the message, through encryption, the original message is transformed in a more random looking form, that of a ciphertext. This way, the statistical impact of the embedding process is drastically reduced because the randomness of the least significant bits is not altered heavily. In *Figure 5.3* a general image pixel array is presented.

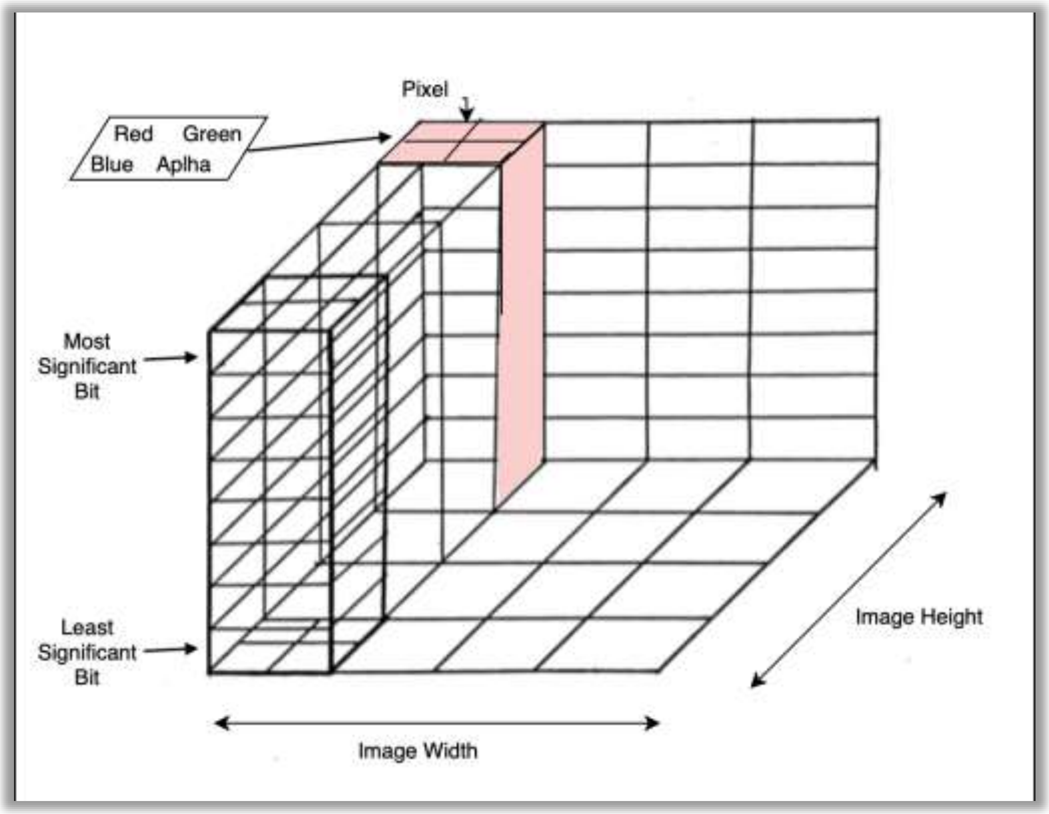


Figure 5.3 - Image pixel array

If we take an example the secret message “A” would first be encrypted using the recipients public key. The original pixel data is represented in *Figure 5.4*. The result is a 1024-bit long cipher text as seen bellow.

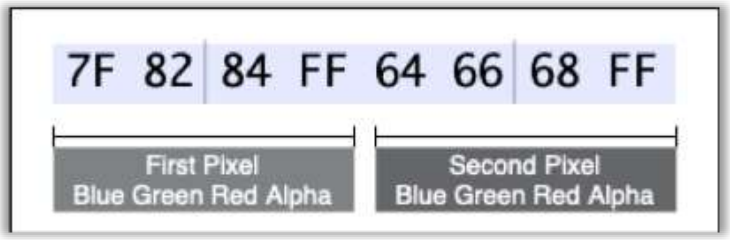


Figure 5.4 - Two pixel example

“A” – 01000001 (binary format)

Ciphertext - 00010110 11100100 01110001 …

Original first pixels

01111111 10000010 10000100 11111111 01100100 01100110 01101000 11111111

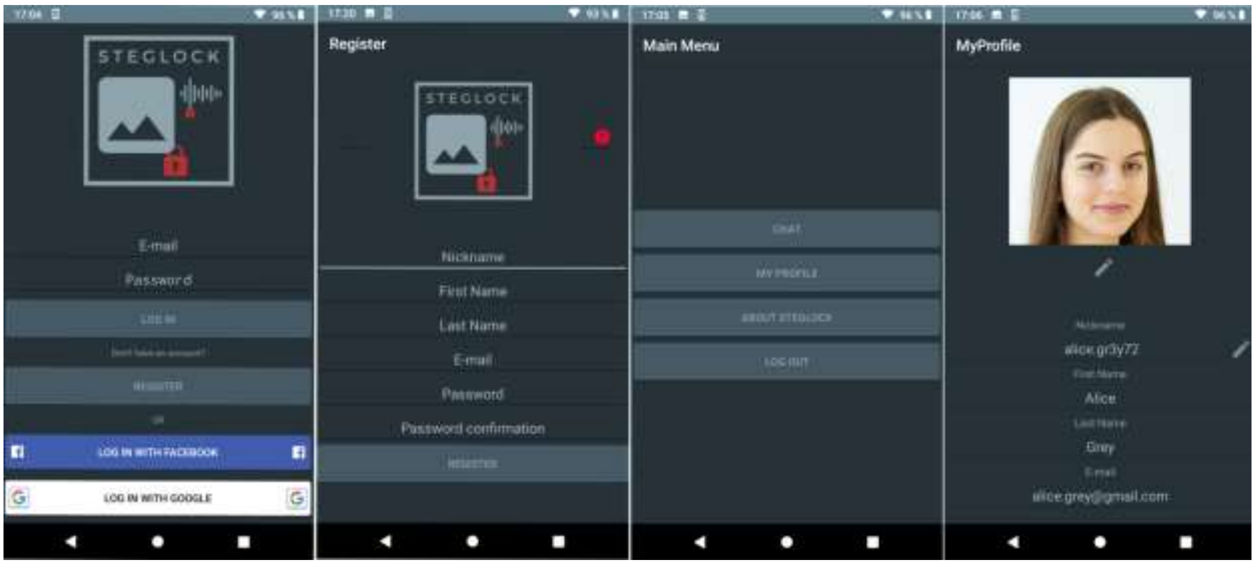
Altered first pixels

01111110 10000010 10000100 11111111 01100100 01100111 01101001 11111110

6. Application usage

On running the application, the login screen is displayed in which the user should input his credentials, can authenticate using social accounts or may begin registering a new account in the case of a new user. If the user is new he has to fill in a registration form.

After successful authentication, the user is presented the main menu of the application which allows different actions, like chatting, visualizing and eventually modifying profile information and logging out. By pressing the chat button, the user can begin sending messages. *Figure 6.1* presents how the Login, Register, Menu and Profile activities are looking like in the mobile application.



Nickname
alice.gr3y72
First Name
Alice
Last Name
Grey
E-mail
alice.grey@gmail.com

Figure 6.1 - Log in, Register, Menu and Profile Activities

After pressing the chat button of the main menu, the user is presented an activity consisting of two tabs, the chats tab and the users tab. The chats tab displays a list of the current conversations the user has with others. The users tab displays a list of all the registered accounts and permits searching for a specific user by nickname. A press on a nickname in the chats tab or in the users tab will open the existing or new conversation, respectively.

The Message activity displays the different types of messages sent and received by users inside rounded boxes on the left or right of the screen depending on who sent the message and has different buttons and controls which allows the user to send different types of messages.

If the image button is pressed, the application displays the device gallery for the user to choose a picture to send. After the picture is chosen the Send Stego-Image Activity is presented allowing the user to input a secret message to be hidden inside the picture. Similarly, if the photo button is pressed the camera is opened for the user to take a photo which will be used as the cover or if the recording button is pressed, the user is presented the Send Stego-Recording Activity, which allows the user to record audio sounds as the cover file for the steganographic process. In *Figure 6.2* the Chat, Message, Stego-Image and Stego-Recording Activities are presented.

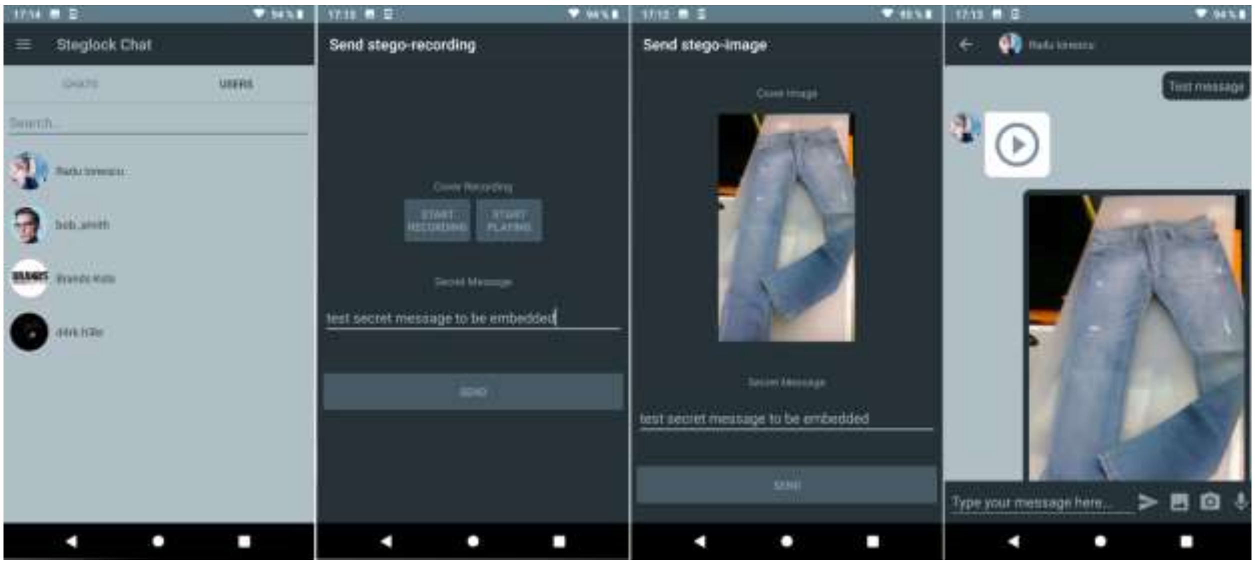


Figure 6.2 - Users, Stego-Recording, Stego-Image and Messages Activities

7. Steganographic output analysis

To test the quality of the application the steganographic output, which is represented by the sent images, photos or sound recordings, that contain embedded secret text messages, should be analyzed from different points of view. The statistical approach of analyzing stego-objects is called steganalysis.

The main goal of steganalysis is, starting from a set of digital files, to conclude if some of the files contain embedded information inside or not, and which files contain this type of information. In general, steganalysis may only detect the presence of the added data, but cannot extract the hidden information.

Generally, steganography can be easily detected by steganalysis, by detecting the modification of the statistical properties of the file. There are many kinds of steganalysis and trying to trick the statistical approaches is pretty hard and attention needs to be paid when developing an as secure as possible steganographic schema.

Another way of detecting steganography, a more straightforward one, is comparing the steganographic object with the original object. In the hypothesis of using a Least Significant Bit steganographic algorithm, if we take the values of all the pixels in the steganographic object and deduct them from the values of the original image pixels, if the result is different than zero for any pixel, we can conclude that the original file has been altered.

The StegLock mobile applications steganographic outputs, as other similar mobile applications outputs, can be pretty effortlessly debunked. In case the steganalysis finds which files contain added information, even if the hidden data is recovered from the steganographic object, it is encrypted using a symmetrical encryption algorithm, so the secret information embedded is relatively safe. The quality of the steganographic output of the developed mobile application is not greater than the general quality of other similar solutions.

8. Conclusions and recommendations

In conclusion, steganographic techniques are very interesting and were used for centuries and are also used today, in their digital form. In my opinion, the quality of the outputs of this type of algorithms should be analyzed rigorously to make sure they are as safe as possible against steganalysis techniques. Statistical approaches may also be used to determine the secure steganographic capacity of the files, which is the maximum amount of information that can be hidden into a file, without considerably changing the statistical

Other similar processes tightly coupled with steganography are fingerprinting and digital watermarking. A fingerprinting algorithm is used to generate a unique mark for a piece of data and embed it into that specific file. This is very useful when you want to supply some files and protect them from ongoing distribution. Watermarking also embeds a mark of the files with the purpose of signifying ownership.

As opposed to steganography, in fingerprinting and watermarking the existence of the embedded data is publicly known, whereas steganography tries to completely hide that there is any information hidden inside. While a successful attack concerning a watermarking or fingerprinting algorithm consists of removing the watermark or fingerprint, basically removing the ownership protection, an attack on a steganographic system should detect and eventually extract the hidden data.

As further development, the steganographic algorithms used by the mobile application can be improved to output object of greater quality, making the analysis of them harder. Also, the amount of data that can be embedded into the multimedia files may be calculated in a more secure manner, for the same reason.

9. Bibliography

[1] R. G. BALDWIN, "Processing Image Pixels Using Java: Controlling Contrast and Brightness," [Online]. Available: https://www.developer.com/java/other/article.php/3441391. [Accessed 12 March 2020].

[2] R. G. BALDWIN, "Processing Image Pixels using Java, Getting Started," [Online]. Available: https://www.developer.com/java/other/article.php/3403921. [Accessed 12 March 2020].

[3] S. K. ARORA, "Audio Steganography : The art of hiding secrets within earshot (part 2 of 2)," [Online]. Available: https://medium.com/@sumit.arora/audio-steganography-the-art-of-hiding-secrets-within-earshot-part-2-of-2-c76b1be719b3. [Accessed 11 March 2020].

[4] S. K. ARORA, "Audio Steganography : The art of hiding secrets within earshot (part 1 of 2)," [Online]. Available: https://medium.com/@sumit.arora/audio-steganography-the-art-of-hiding-secrets-within-earshot-part-1-of-2-6a3bbd706e15. [Accessed 11 March 2020].

[5] M. S. SHAHREZA, "Stealth Steganography in SMS," *International Conference on Wireless and Optical Communications Networks,* 2006.

[6] M. S. MEGHA, "Methods of Audio Steganography," *International Journal of Engineering and Management Research,* vol. 4, no. 3, pp. 154-156, 2014.

[7] M. ZAMANI, A. MANAF, R. AHMAD, F. JARYANI, H. TAHERDOOST and A. M. ZEKI, "A Secure Audio Steganography Approach," *International Conference for Internet Technology and Secured Transactions,* 2009.

[8] N. HAMID, A. YAHYA, B. AHMAD and O. M. AL-QERSHI, "Image Steganography Techniques: An Overview," *International Journal of Computer Science and Security,* vol. 6, no. 3, pp. 168-187, 2012.

[9] R. CHANDRAMOULI and N. MEMON, "Analysis of LSB based image steganography techniques," *Proceedings 2001 International Conference on Image Processing,* vol. 3, pp. 1019-1022, 2001.

[10] N. F. JOHNSON, Z. DURIC and S. JAJODIA, Information Hiding: Steganography and Watermarking - Attacks and Countermeasures, Boston, MA: Springer, 2001.

[11] C. MAITI, D. BAKSI, I. ZAMIDER, P. GORAI and D. R. KISKU, "Data Hiding in Images Using Some Efficient Steganography Techniques," *Communications in Computer and Information Science,* vol. 260, pp. 1-9, 2011.

[12] P. POCATILU, I. IVAN, A. VIȘOIU, F. ALECU, A. ZAMFIROIU and B. IANCU, Programarea Aplicațiilor Android, Bucharest: Editura ASE, 2015.

[13] N. PROVOS and P. HONEYMAN, "Hide and Seek: An Introduction to Steganography," *IEEE Security and Privacy Journal,* vol. 1, pp. 32-44, 2011.

[14] M. FORTRINI, "Steganography and digital watermarking: A global view.," 2011.

[15] J. H. P. ELOFF, T. MORKEL and M. S. OLIVIER, "An Overview Of Image Steganography," *Proceedings of the Fifth Annual Information Security South Africa Conference,* 2005.

[16] "Documentation | Android Developers," [Online]. Available: https://developer.android.com/docs. [Accessed 10 March 2020].

[17] "Steganography," [Online]. Available: https://en.wikipedia.org/wiki/Steganography. [Accessed 9 March 2020].

[18] "What is Steganography? Webopedia Definition," [Online]. Available: https://www.webopedia.com/TERM/S/steganography.html. [Accessed 25 April 2020].

[19] N. EL-EMAM, "Hiding a Large Amount of Data with High Security Using Steganography Algorithm," *Journal of Computer Science,* vol. 3, pp. 223-232, 2007.

[20] "Google Firebase Documentation," Google, [Online]. Available: https://firebase.google.com/docs. [Accessed 21 March 2020].

[21] R. G. BALDWIN, "Steganography 101 using Java," [Online]. Available: https://www.developer.com/java/ent/article.php/10933_3530866_2/Steganography-101-using-Java.htm. [Accessed 12 March 2020].

[22] R. J. ANDERSON and F. A. P. PETITCOLAS, "On The Limits of Steganography," *IEEE Journal of Selected Areas in Communications,* vol. 16, no. 4, pp. 474-481, 1998.

[23] D. ARTZ, "Digital steganography: hiding data within data," *IEEE Internet Computing,* vol. 5, no. 3, pp. 75-80, 2001.

[24] N. F. JHONSON and S. JAJODIA, "Exploring steganography: Seeing the unseen," *Computer,* vol. 31, no. 2, pp. 26-34, 1998.

[25] W. BENDER, D. GRUHL, N. MORIMOTO and A. LU, "Techniques for data hiding," *IBM Systems Journal,* vol. 35, no. 3.4, pp. 313-336, 1996.