# Project Report

# on

# “SECURIFY”

# Submitted

# To

# Information Technology Department

# June 2020

# 

# Dr. S. & S. S. Ghandhy College of Engineering & Technology, Surat

# Gujarat Technological University

# Project Report

# on

# “SECURIFY”

Developed By:

BHALANI KAHAN DEPPAKUMAR (176120316003)

SHETA VIVEK KANTIBHAI (176120316051)

TARSARIYA DHAVAL DINESHBHAI (176120316059)

Guided By:

MRS. BHAVINI S. PATEL (Lecturer in I.T.)

A project report submitted to Gujarat Technological University in partial fulfilment of the requirements for Diploma Engineering in Information Technology

# June 2020

# 

# Dr. S. & S. S. Ghandhy College of Engineering & Technology, Surat

# Gujarat Technological University

# Related image

CERTIFICATE

This is to certify that project work embodied in this report entitled “**Securify”** was carried out by **Mr.** **Bhalani Kahan Deepakkumar (176120316003)**, **Mr. Sheta Vivek Kantibhai (176120316051)** and **Mr. Tarsariya Dhaval Dineshbhai** **(176120316059)** at **Dr. S & S. S. Ghandhy College of Engineering & Technology, Surat.**

For partial successful completion of **Project-II (3361606)** of **Semester-6** Diploma engineering in Information Technology to be awarded by Gujarat Technological University. This project work has been carried out under my supervision and is to the satisfaction of the department.

**Date:**

**Place:** Surat

**GUIDE H.O.D. PRINCIPAL EXTERNAL EXAMINER**

DECLARATION OF ORIGINALITY

We hereby certify that we are the sole author of this project and that neither any part of this work nor the whole work has been submitted for a degree to any other University or Institution.

We certify that, to the best of our knowledge, our work does not infringe upon anyone’s copyright nor violate any proprietary rights and that any ideas, techniques, quotation, or any other material from the work of other people included in our report or otherwise, are fully acknowledged in the accordance with the standard referencing practices.

We declare that this is a true copy our report, including any final revisions, as approved by our supervisor.

**Date:**

**Place:** Surat

**Yours Sincerely,**

Bhalani Kahan Deepkkumar (176120316003)

Sheta Vivek Kantibhai (176120316051)

Tarsariya Dhaval Dineshbhai (176120316059)

ACKNOWLEDGEMENT

The success and final outcome of this project required a lot of guidance and assistance from many people and we are extremely privileged to have got this all along the completion of our project. Many peoples have contributed to this project in variety of ways.

We heartily thank our project guide, **Mrs. Bhavini S. Patel** for his guidance and suggestions during this project work. We would not forget to remember **Mr. Jaydeep Mahida** for his encouragement and more over for his timely support and guidance till the completion of our project work. The project would not have been successfully completed without continuous support, motivation extended by our colleagues and friends who were always with us whenever we needed.

We are thankful to and fortunate enough to get constant encouragement, support and guidance from all Teaching staffs of **Information Technology Department** which helped us in successfully completing our project work.

Last but not least, thanks to **Dr. S. & S. S. Ghandhy College of Engineering & Technology** for providing us the platform to represent the project.

**Yours Sincerely,**

Bhalani Kahan Deepakkumar (176120316003)

Sheta Vivek Kantibhai (176120316051)

Tarsariya Dhaval Dineshbhai (176120316059)

# Abstract

In real world, there are many crime cases occurring like molestation, kidnapping, and other emergencies such as fire & medical issues. **“Securify (Security Notifier)”** is a mobile application which can provide immediate assistance to the person in emergency by alerting the relatives or any ICE (In-case of emergency) responder by just pressing the hardware buttons of the mobile.

# Project Definition

By using “**Securify”**, we don’t need to open the app and send the alert. We can do so by just pressing the hardware buttons of the mobile. So, we can send the alert as fast as possible and get help.

The app will ensure to provide an easy to use mechanism to send the alert through SMS, by pressing the hardware buttons of the smartphone. We can easily locate the person in need, as the alert also includes the current location.

If the user thinks that something wrong or illegal is happening, then the app also allows to start multimedia services (like video recording), by just pressing the hardware buttons.

# TABLE OF CONTENTS

# 

1. Introduction 1

1.1 Specification 1

1.2 Overview of Language 2

1.3 Scope 3

2. Requirements Analysis 4

2.1 Requirements Analysis 4

2.2 Problem Solving Techniques 5

2.3 Project Life Cycle Model 6

2.4 Software Requirements Specification 8

3. System Design 15

3.1 Organization Chart 15

3.2 ER Diagram 16

3.3 Data Flow Diagram 17

3.4 Data Dictionary 19

4. UML 20

4.1 Use Case Diagram 20

4.2 Sequence Diagram 22

4.3 Class Diagram 23

4.4 Activity Diagram 24

5. GUI/Application Interface 25

6. Testing 28

7. Conclusion 30

Bibliography 32

Future Work 31

# List of Figures

Figure 2.3 Scrum Framework 7

Figure 3.1 Organization Chart 15

Figure 3.2 ER Diagram 16

Figure 3.3.1 DFD – Level 0 – Context Diagram 17

Figure 3.3.2 DFD – Level 1 18

Figure 4.1 Use Case Diagram 21

Figure 4.2 Sequence Diagram 23

Figure 4.3 Class Diagram 24

Figure 4.4 Activity Diagram 25

5 User Interface screenshots 26

# List of Tables

Table 2.4.1 Abbreviations for SRS 8

Table 3.4.1 User 19

Table 3.4.2 Volunteer 19

Table 3.4.3 Alert 19

5 Test Cases 28

# Chapter 1. Introduction

1.1 Specification

*Development*

Our development workstation will have the following specification:

* **Software Specification**
  + **IDE**: Android Studio v3.5
  + **Language**: Java, XML
  + **Platform**: Microsoft Windows 10
  + **Backend Service**: Google Firebase
* **Hardware Specification**
  + **Processor**: Intel i5-8th Gen
  + **RAM**: 4GB or more
  + **Testing Equipment**: Android Smartphone

*End-User*

The end-user must have an Android smartphone with following specification:

* **Platform**: Android Lollipop (API 21 or later) [1]
* GPS (Location)
* Internet connection
* A working SMS plan

1.2 Overview of Language

“Securify” will be developed using the native Android SDK. It is the most native way of developing Android applications. We are targeting the Android devices as most of the people are using an Android smartphone right now.

We will use *Java* and *XML* to develop the application.

*Java*

Java will be used to design the logic structure of the whole Android application. There will be a global Application class which contains various Android components like

* Activities (Visible UI)
* Services (Background Tasks)
* Broadcast Receivers
* Content Providers [2]

By writing the Java code, the app will interact with the various UI elements which statically defined in **XML**.

*XML*

XML will provide the static UI (User Interface) written in separate ***.xml*** files. These designs will be stored as separate resources will be a part of the Android package. [3]

Most of the implementation will be done using the core Android SDK, along with some additional open-source libraries for licensing. We will use the latest Android SDK available (currently ***API 28***) [1] for development, as it provides new features and more secure applications.

1.3 Scope

“Securify” will provide the following features:

* By pressing a sequence of volume buttons, we can:
  + Send an alert without opening SMS or dialer app
  + Call an ICE responder or the contact
  + Start video or voice recording
* A dialog box will be shown to perform all operations, without opening the app
* Up to 5 contacts can be selected which will be notified through alert
* The current location of the victim will also be sent
* The volunteer can easily send an acknowledgement or a customized reply again from a simple dialog box.

# Chapter 2. Requirements Analysis

2.1 Requirements Analysis

We performed some brainstorming sessions, and gathered the following requirements:

* Send an alert signal to selected contacts by pressing a sequence of Volume button.
* A confirmation dialog box will be shown from where the signal will be sent. In this dialog box, we can specify the following things:
* Header
* Customizable description
* Current location
* Start video or voice recording
* Call one of the selected contacts
* Call an ICE responder
* **4th** and **5th** can be toggled to turn the visibility in the dialog box.
* The alert can also be sent by opening the app manually.
* If the alert is successfully delivered to the receiver, the sender will get a haptic feedback.
* The receiver will get a notification on receiving the message and a dialog box which displays:
* Alert message
* Current location of the sender
* A button for sending an acknowledgement to the sender.
* Call back the sender
* Call an ICE responder
* Maximum five contacts can be selected for alert.
* The alert will be sent through SMS and Internet.
* We can customize whether to call the contact or send an alert message.
* The app will also display an **Alert Log** about all the alerts sent and calls made.
* The instant alert service (through volume buttons) can also be scheduled to prevent unintended volume button presses.
* When the service is running and a valid sequence is pressed, the system volume won't be affected.

2.2 Problem Solving Techniques

To develop “Securify”, we will use the bottom-up (object oriented) approach as our problem-solving technique. The two main reasons why we chose this approach are:

* The Android SDK is completely based on Java which is completely object oriented. The whole Android framework is a collection of classes.
* Nowadays, the procedure-oriented or top-down approach is getting obsolete. It becomes easy to visualize a problem as a collection of real-world objects. When developing a complex system, such as the User Interface in case of Android™, we can think of every drawable as an object, and then using the same baseline for deriving new views.

2.3 Project Life Cycle Model

For developing “Securify”, we think that agile development will be the most suitable. The requirements, in the context of our app, needs to be delivered as soon as possible, and then there are some features which aren't necessary in accordance to our primary scope.

The primary target would be to **send the alert message so that we can get help ASAP**. So, we can develop our app in increments, with our primary target as the first increment.

Moreover, the requirements involve a considerable amount of real-life analysis, as the technology is evolving at a rapid phase and the security restrictions that Google is imposing on Android™.

Looking at this rapid evolution, our main focus should be to deliver the product while adapting with the evolving technology and security.

So, we think that the best way to develop “Securify” is to go **agile**. In agile, we follow the four basic principles [4]:

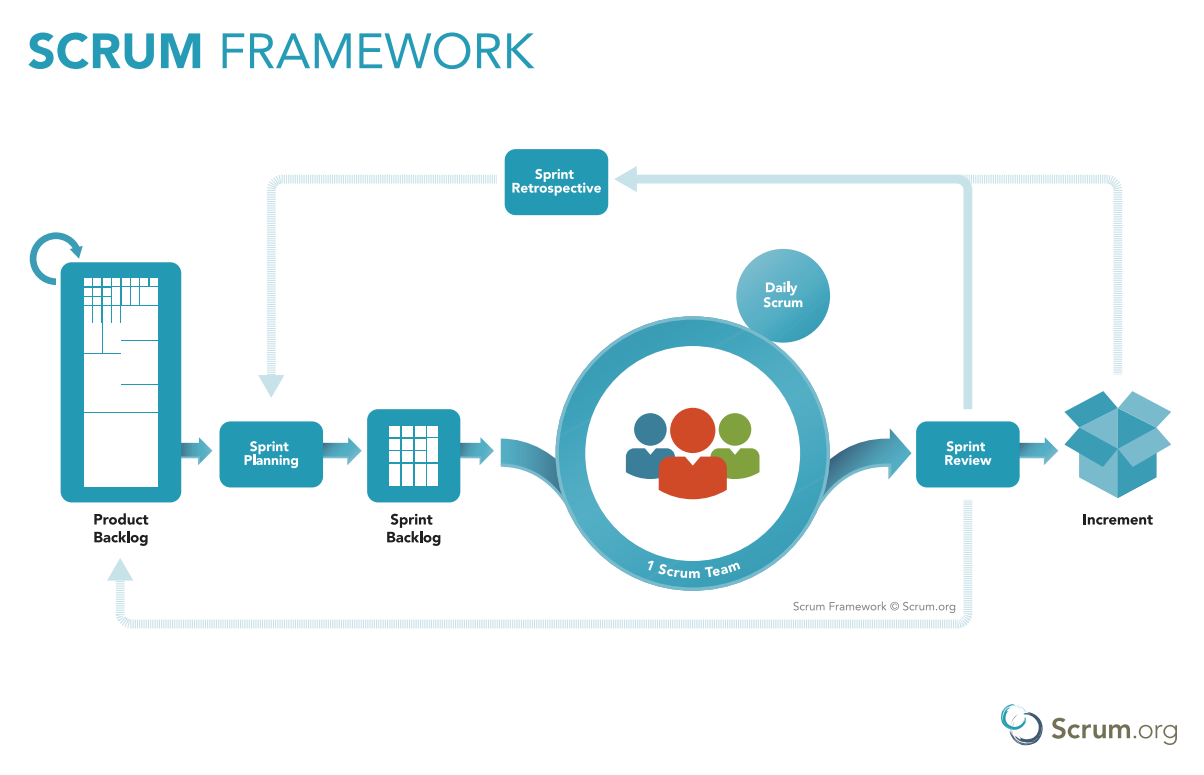
* Individuals and interactions over processes and tools
* Working software over comprehensive documentation
* Customer collaboration over contract negotiation
* Responding to change over following a plan

*Why Scrum?*

We will be using **Scrum** as our agile development model. Scrum is one of the popular SDLC model and it meets our development criteria of being agile.

Scrum is a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value [5].

The following figure shows a brief overview of how Scrum works:

Figure 2.3.1 Scrum Framework

2.4 Software Requirements Specification

The table 2.4.1 shows some abbreviations related to our requirements.

Table 2.4.1 Abbreviations for SRS

|  |  |
| --- | --- |
| ****Term**** | ****Definition**** |
| **Victim** | **The user who is in emergency and the one sending the alert** |
| **Volunteer** | **The user who is selected by the victim and the one receiving the alert** |
| **VIFR** | **Victim’s functional requirement** |
| **VOFR** | **Volunteer’s functional requirement** |
| **QR** | **Quality Requirement** |
| **FR** | **Common functional requirement** |
| **SR** | **Security Requirement** |
| **User** | **Victim or Volunteer** |
| **TITLE** | **Title of the requirement** |
| **DESCRIPTION** | **Description about the requirement** |
| **DEPS** | **Any dependencies to another requirement** |

*Common Requirements*

**This section includes all the requirements which are common among Victim and Volunteer.**

****FR – 1****

**TITLE: Create a new account**

**DESCRIPTION: When the app is opened for first time, the user should be able to register for a new Securify account with following information:**

* **First Name & Last Name**
* **Mobile No.**
* **Password (includes uppercase, lowercase letters and digits)**
* **Email**

****FR – 2****

**TITLE: Login**

**DESCRIPTION: When the app is opened for first time, the user should be able to login with its Securify account using one of the following information:**

* **Email & Password**
* **Mobile No. & OTP**

****FR – 3****

**TITLE: Reset password**

**DESCRIPTION: If the user forgets the password, then app should provide reset password facility using OTP verification.**

****FR – 4****

**TITLE: Logout**

**DESCRIPTION: The user should be able to logout from the app and then register or login again.**

****FR – 4****

**TITLE: The app’s home screen should contain a map which shows the user’s current location**

****FR – 5****

**TITLE: On top of the map, an alert button should be displayed**

**DESCRIPTION: On clicking the alert button, the following five options should be displayed:**

* **Call Police**
* **Call Ambulance**
* **Call Fire station**
* **Alert all**
* **Call any custom contact**

****FR – 6****

**TITLE: Edit the profile settings**

****FR – 7****

**TITLE: Display the About & Help section**

****FR – 8****

**TITLE: Toggle whether to type and send the additional when sending the alert.**

****FR – 9****

**TITLE: Toggle whether to turn on voice recording when sending the alert.**

*****Victim’s Functional Requirements*****

**This section includes all the requirements of the Victim.**

****VIFR – 1****

**TITLE: Send an alert to volunteers for help**

**DESCRIPTION: The victim should be able to send an alert to the volunteers through SMS. The alert should fulfil the following requirements:**

****VIFR – 1.1****

**TITLE: Using Volume buttons to send alert**

**DESCRIPTION: On pressing Volume Up/Down button three times, a dialog box should pop up which confirms the alert.**

**If the alert is confirmed, then another dialog box appears, which allows the victim to send some additional message along with the SOS signal.**

****VIFR – 1.2****

**TITLE: Using the app to send alert**

**DESCRIPTION: The alert can also be sent by opening the Securify app and clicking the alert button.**

****VIFR – 2****

**TITLE: Instant alert should not affect the system volume**

**DESCRIPTION: On pressing the volume button three times, the system volume shouldn’t be affected.**

****VIFR – 3****

**TITLE: Schedule the instant alert service**

**DESCRIPTION: The victim should be able to schedule the instant alert service to prevent accidental triggering of instant alert dialog.**

****VIFR – 4****

**TITLE: Manually turn on/off the instant alert service**

****VIFR – 5****

**TITLE: Display a log of alerts**

**DESCRIPTION: The app should display a list of all the alerts sent by the victim with:**

* **Location**
* **Date & Time**
* **Volunteers responded**

****VIFR – 6****

**TITLE: Select the volunteers**

**DESCRIPTION: The victim should be able to select the volunteers which will be alerted through SMS. They should be selected from the contacts of the mobile phone. Maximum 5 contacts should be selected as volunteers.**

****VIFR – 7****

**TITLE: Display all the volunteers**

**DESCRIPTION: The victim should be able to see and modify the volunteers that are selected.**

*****Volunteer’s Functional Requirements*****

**This section includes all the requirements of the Volunteer.**

****VOFR – 1****

**TITLE: Display a dialog on receiving the alert**

**DESCRIPTION: As soon as the volunteer receives the alert, a dialog box should popup displaying the additional message that the victim might have sent along with the SOS signal. The dialog box should fulfil the following additional requirements:**

****VOFR – 1.1****

**TITLE: Send a reply to the victim**

**DESCRIPTION: The dialog box should provide a Reply button which sends an acknowledgement message back to the victim.**

****VOFR – 1.2****

**TITLE: Directly call the victim through the dialog box**

****VOFR – 1.3****

**TITLE: Provide a location button which redirects to the maps application to show the victim’s location.**

****VOFR – 2****

**TITLE: Notify the volunteer on receiving the message.**

**DESCRIPTION: The volunteer’s phone should vibrate and play an app specific notification tone notify the volunteer about the alert message.**

*****Non-functional Requirements*****

* ****Performance Requirements****

****QR – 1****

**TITLE: Instant alert should be fast**

**DESCRIPTION: The instant alert dialog should be fast and not laggy to open up. It is most critical part as it is used for sending the alert.**

****QR – 2****

**TITLE: Fast and smooth app UI**

**DESCRIPTION: The user interface of the app should be fast and smooth enough such that it takes the least possible time to load the various components.**

* ****Security Requirements****

****SR – 1****

**TITLE: The user must be authenticated before using the app**

**DESCRIPTION: The app should not provide the functionality without the user logging in. The authentication should be done either using:**

* **Email & Password**
* **Mobile No. & OTP**

****SR – 2****

**TITLE: The app should provide a privacy policy about user’s data.**

# Chapter 3. System Design

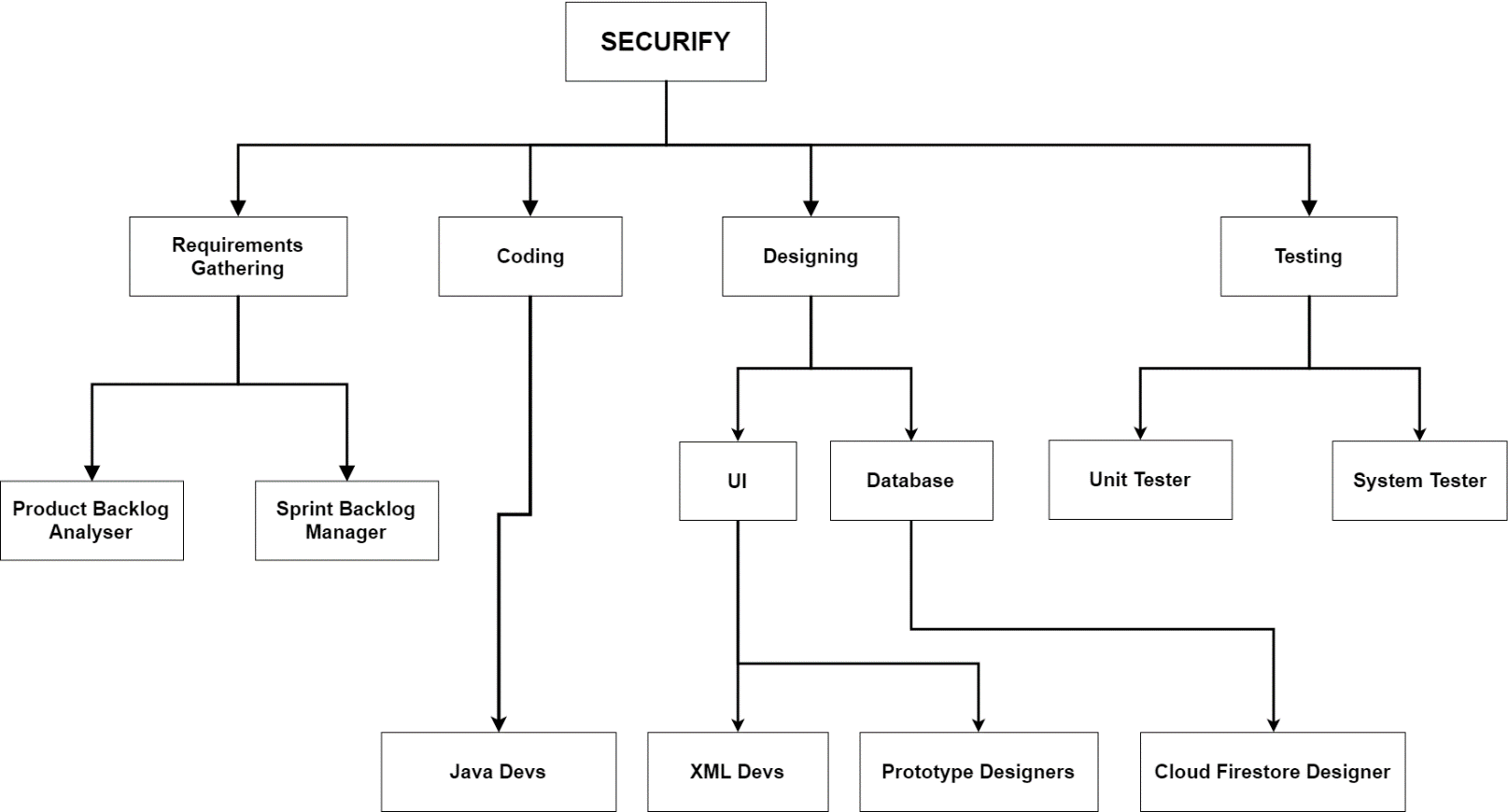
3.1 Organization Chart

Figure 3.1.1 Organization Chart

3.2 ER Diagram

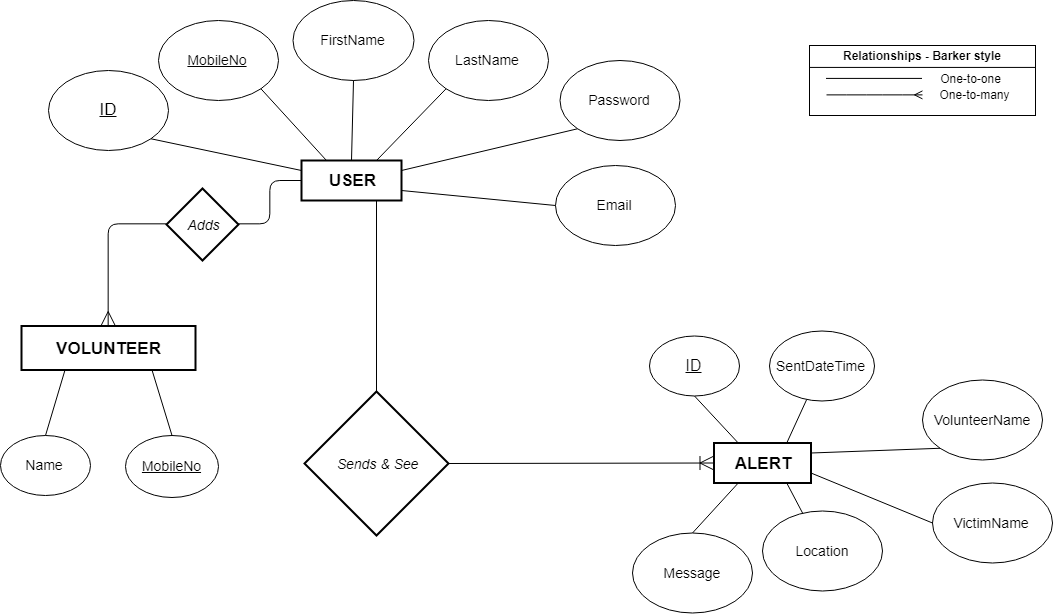


Figure 3.2.1 ER Diagram

3.3 Data Flow Diagram

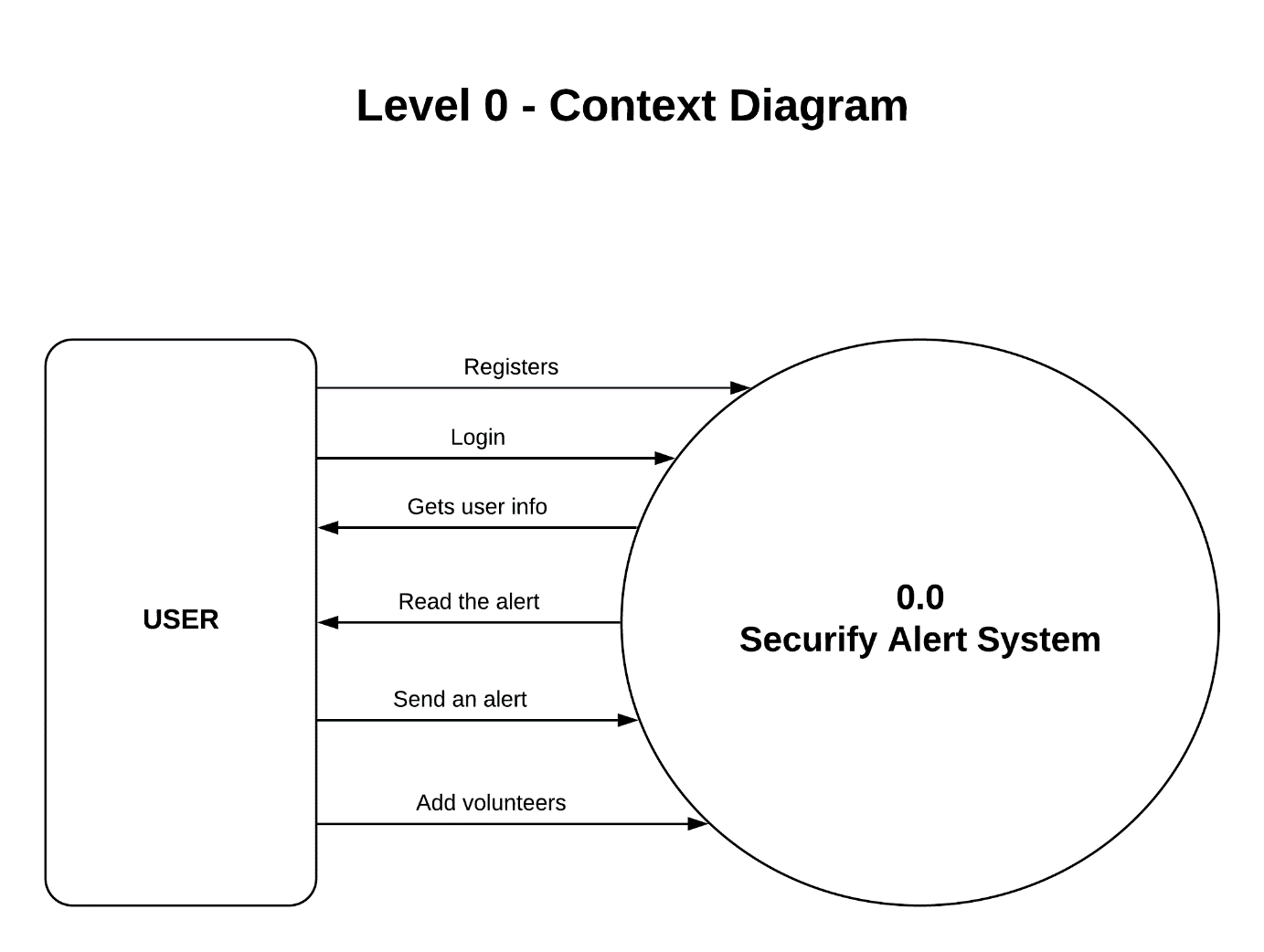


Figure 3.3.1 DFD – Level 0 – Context Diagram

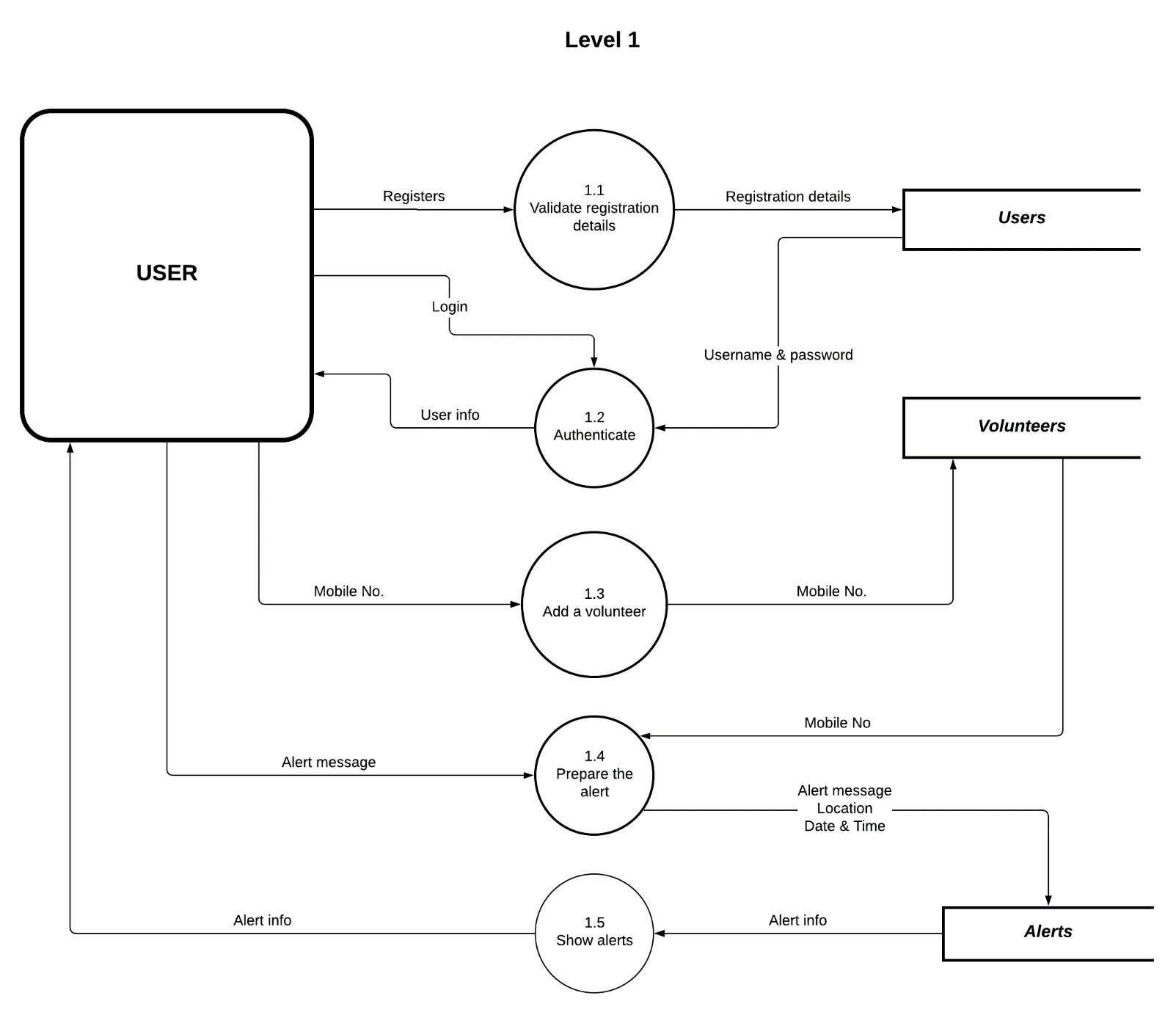


Figure – 3.3.2 DFD – Level 1

3.4 Data Dictionary

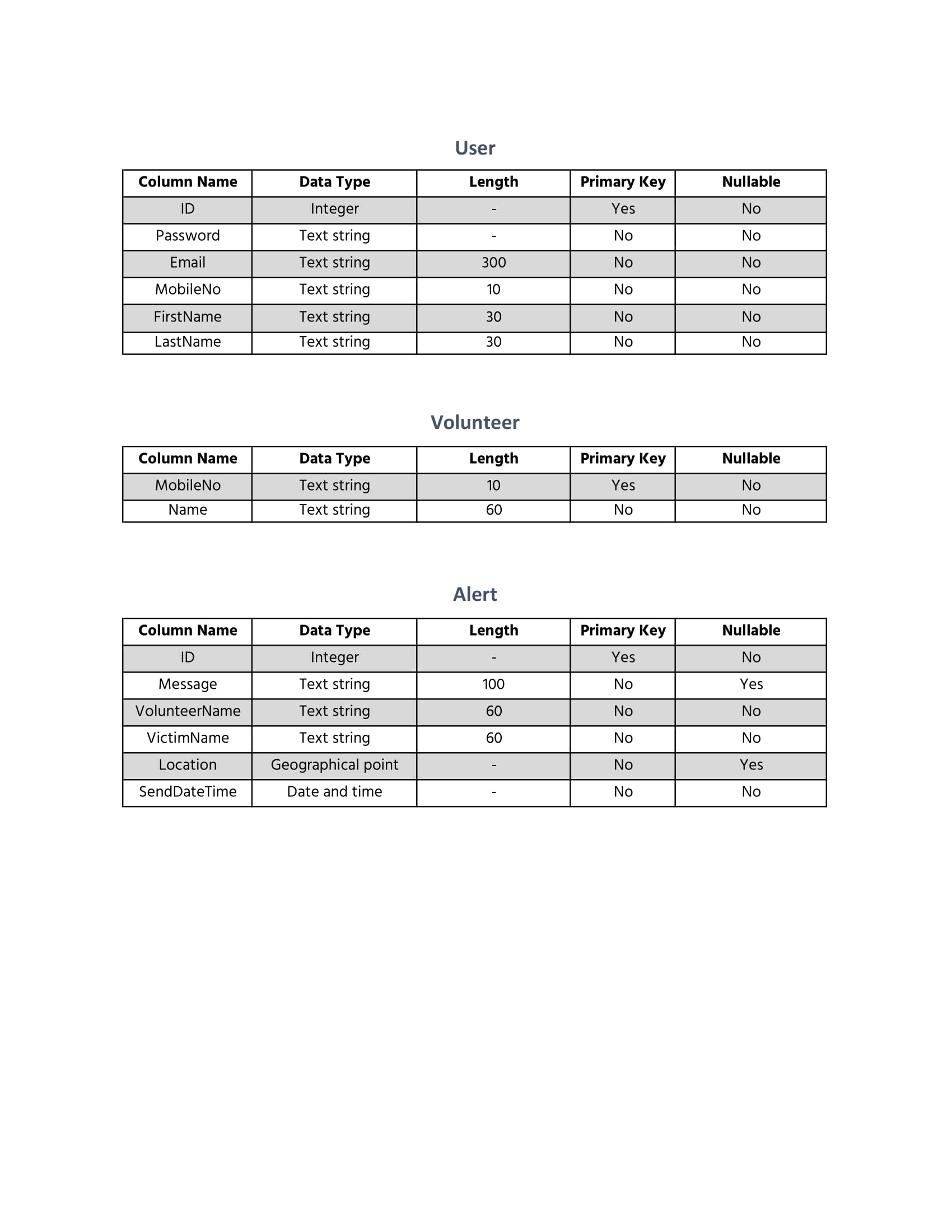
The Figure 3.4.1 shows the various entities of “Securify” and the data dictionary. We are using a NoSQL database provided by Firebase called Cloud Firestore. A **document** in Cloud Firestore is a collection of key-value pairs. The following tables shows the data-types as per the Cloud Firestore specification [6].

Table 3.4.1 User

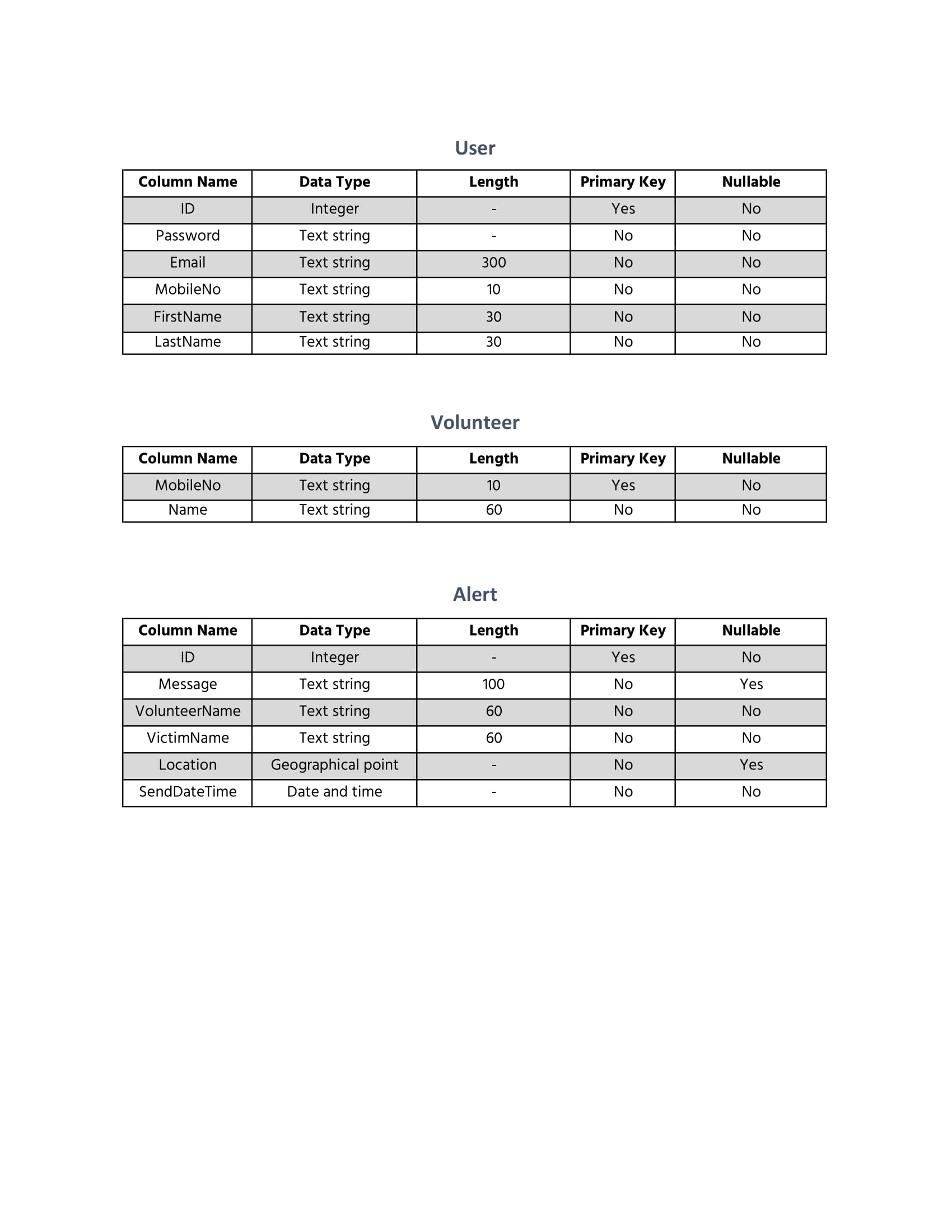


Table 3.4.2 Volunteer

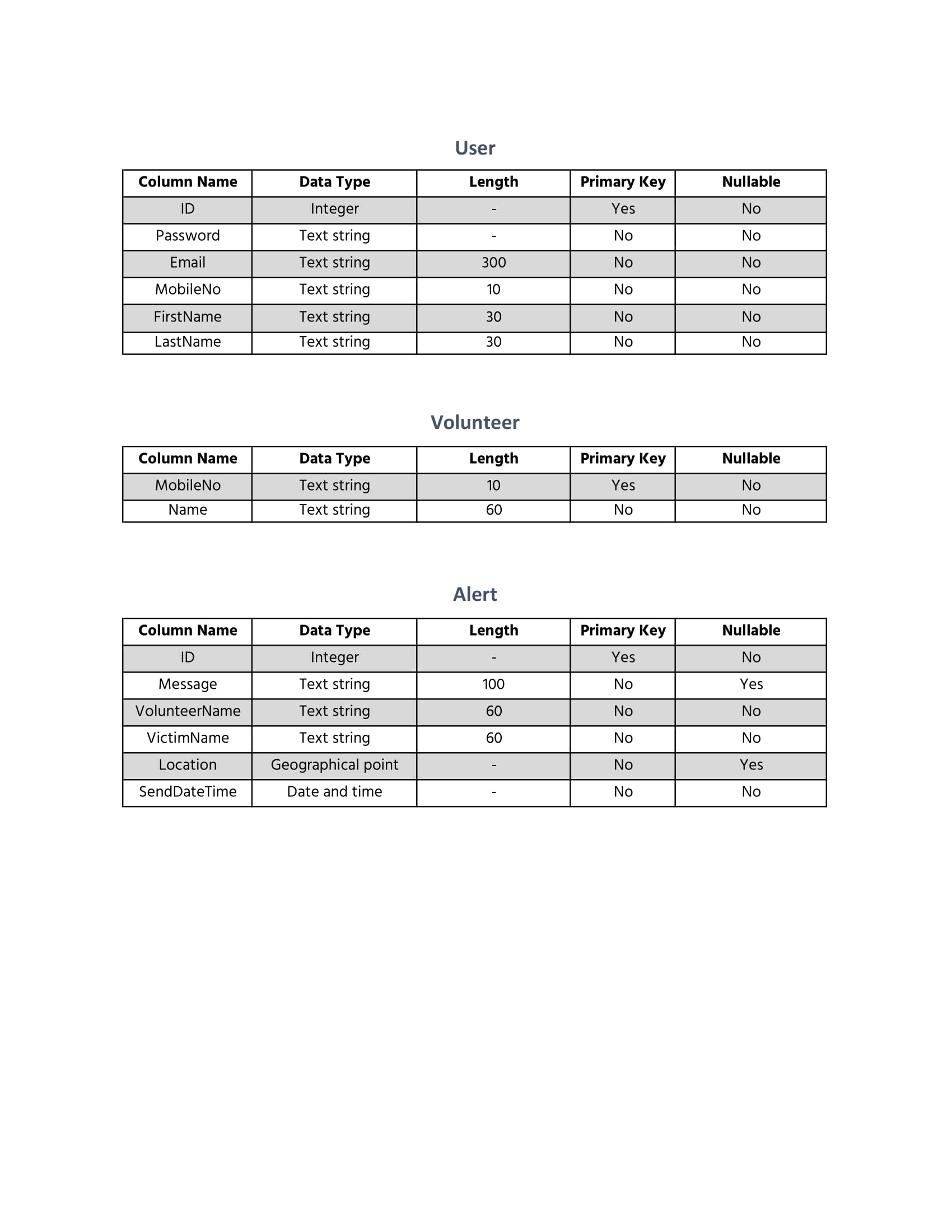


Table 3,4.3 Alert

# Chapter 4. UML

4.1 Use Case Diagram

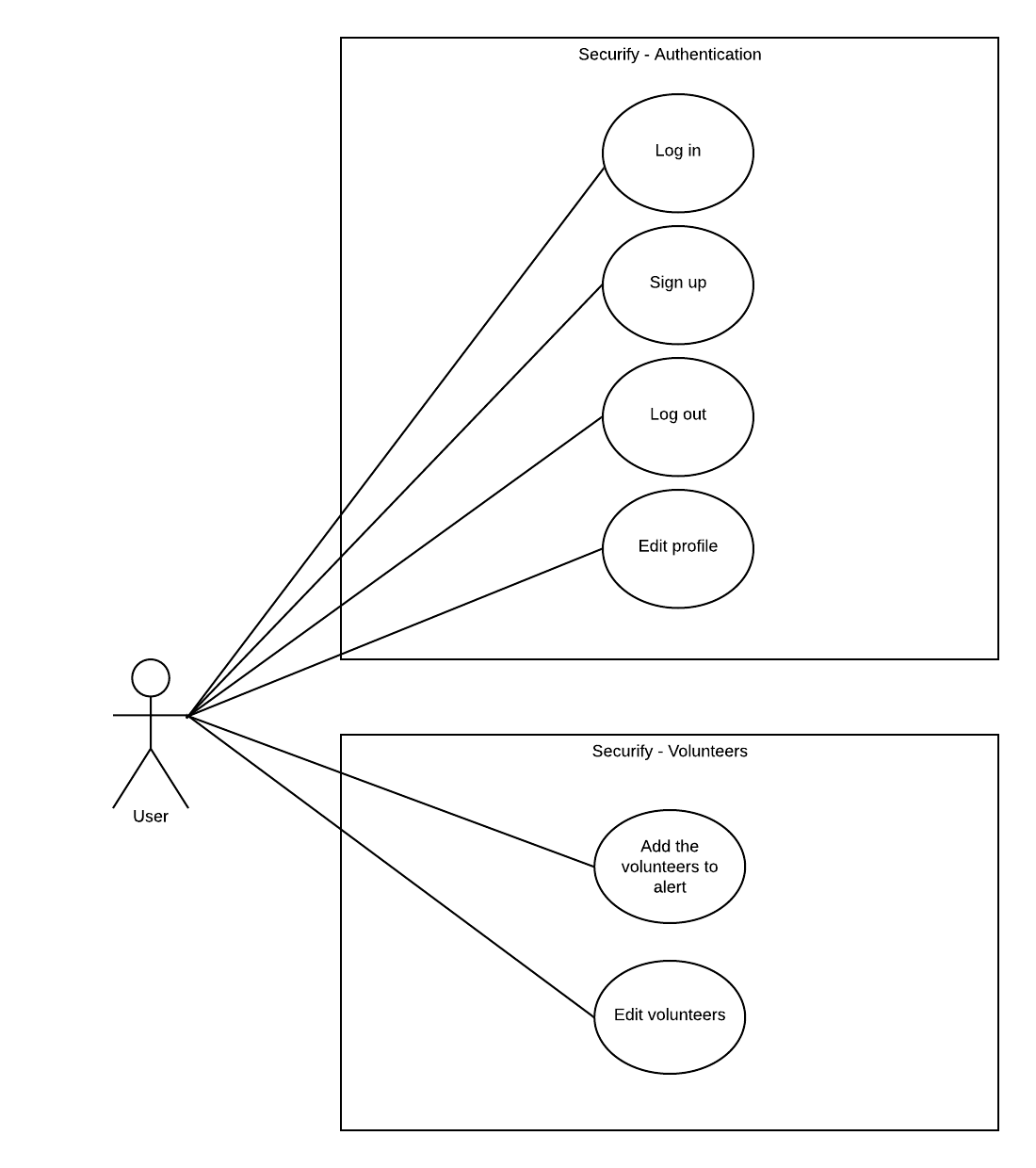


Figure – 4.1.1 Use Case Diagram – Auth & Volunteers

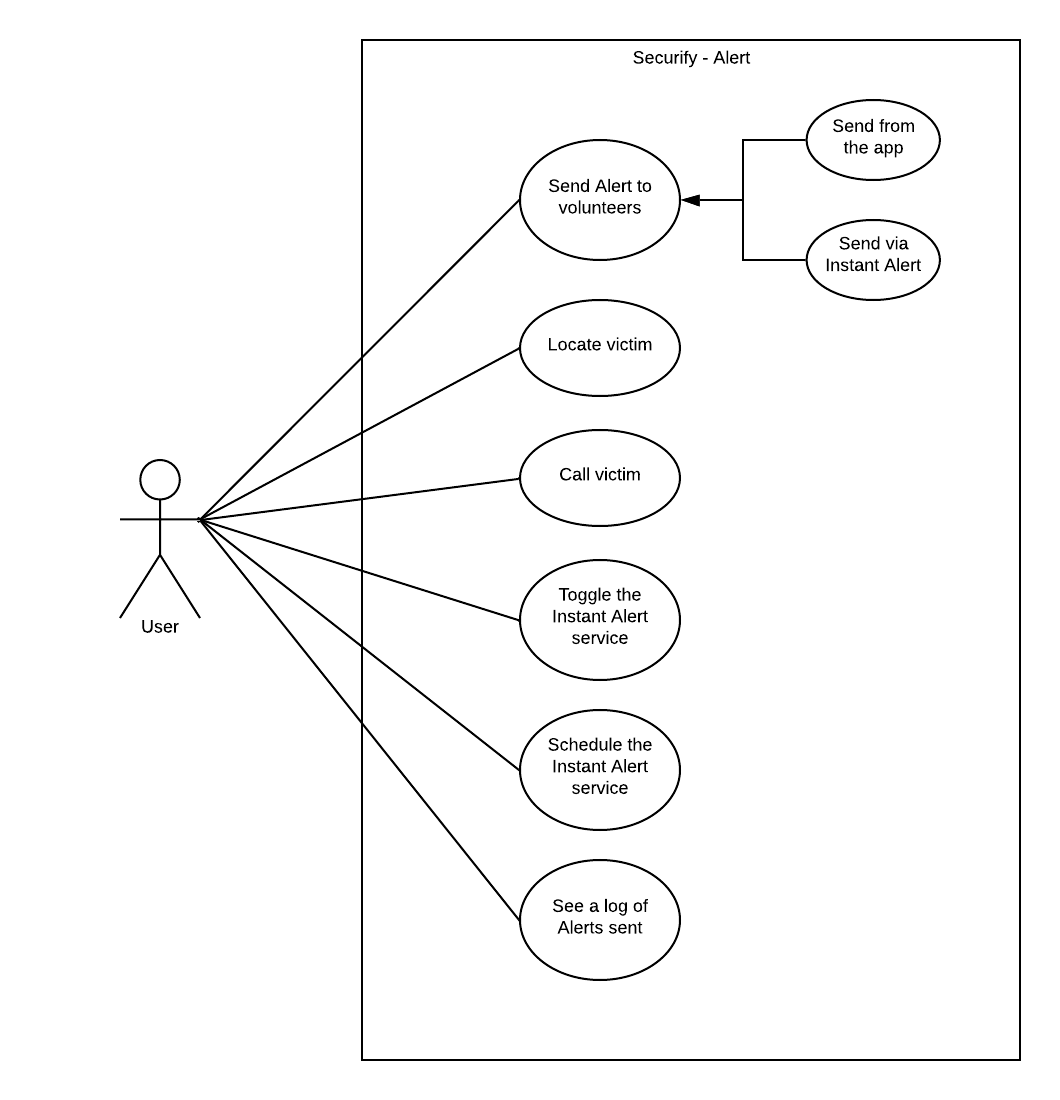
****

Figure – 4.1.2 Use Case Diagram – Alert

4.2 Sequence Diagram

Sending alert

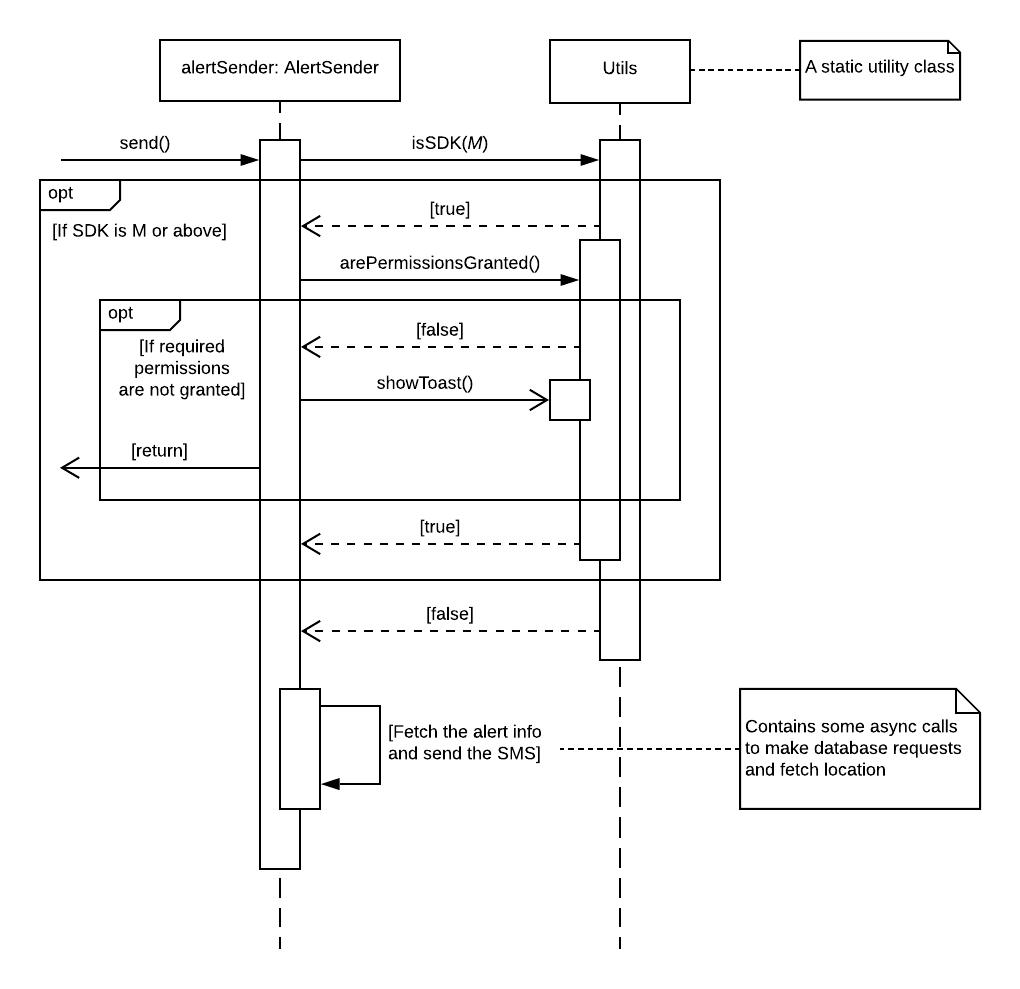


Figure – 4.2.1 Sequence Diagram – Sending alert

4.3 Class Diagram

Alert

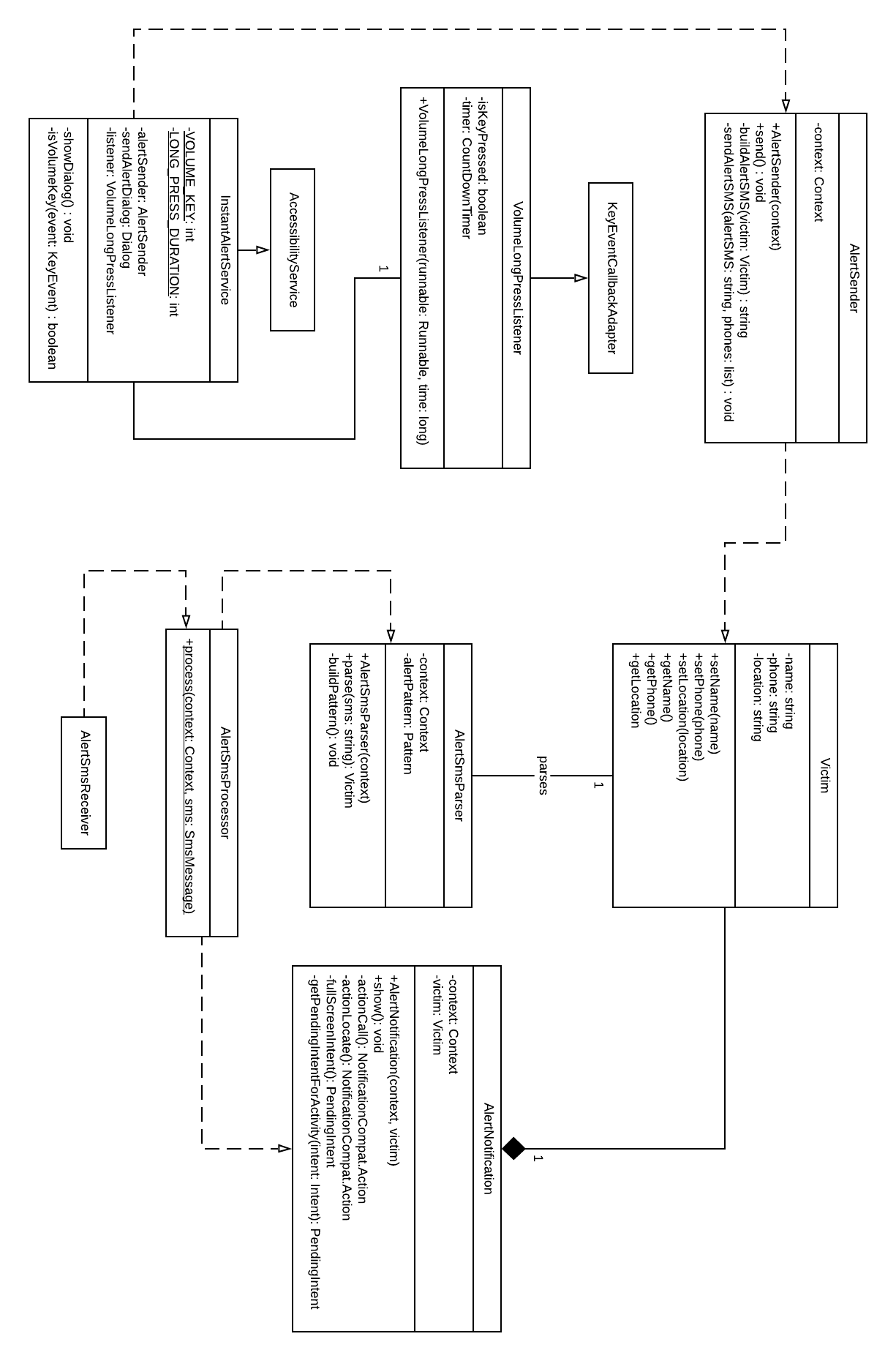


Figure – 4.3.1 Class Diagram – Alert

4.4 Activity Diagram

Sending alert

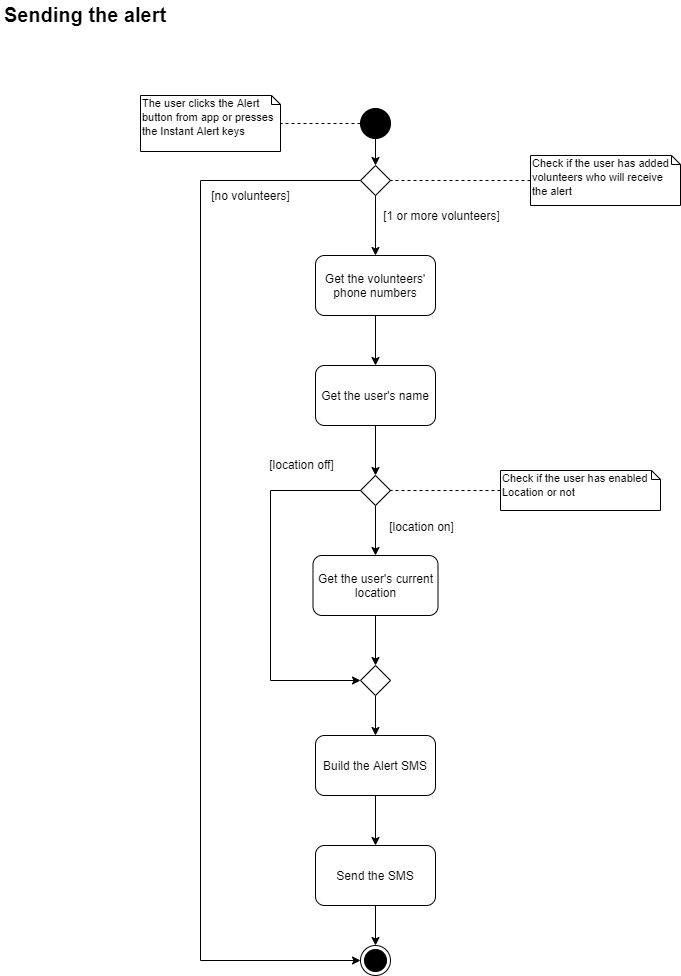
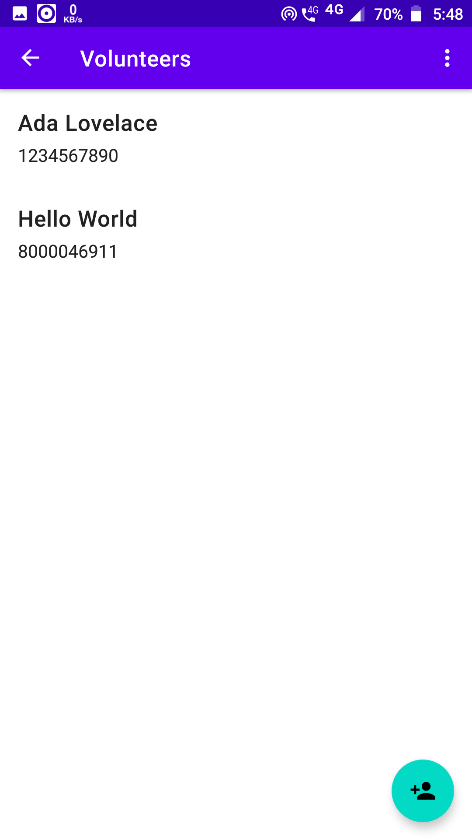
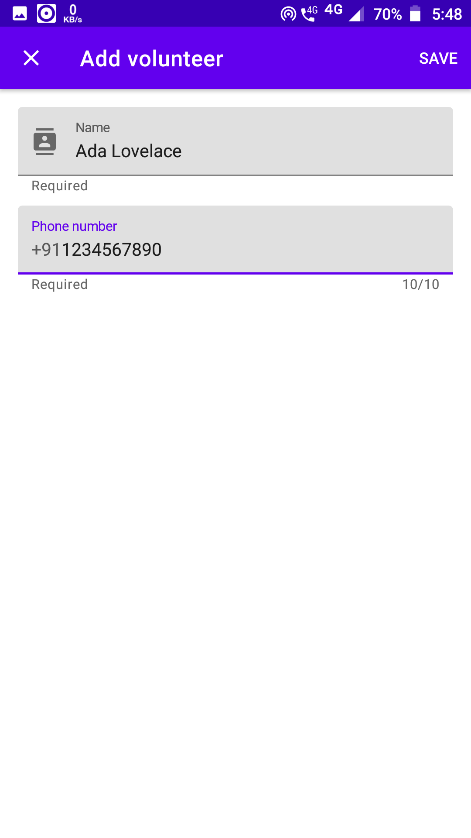
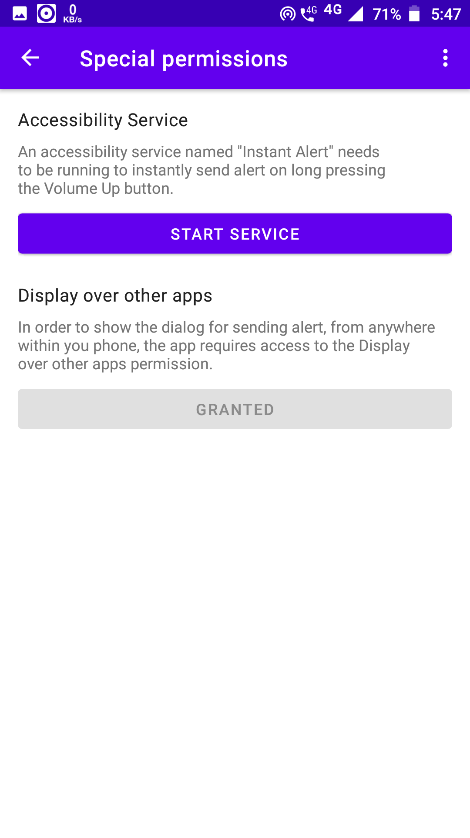


Figure – 4.4.1 Activity Diagram – Sending alert

# Chapter 5. GUI/Application Interface

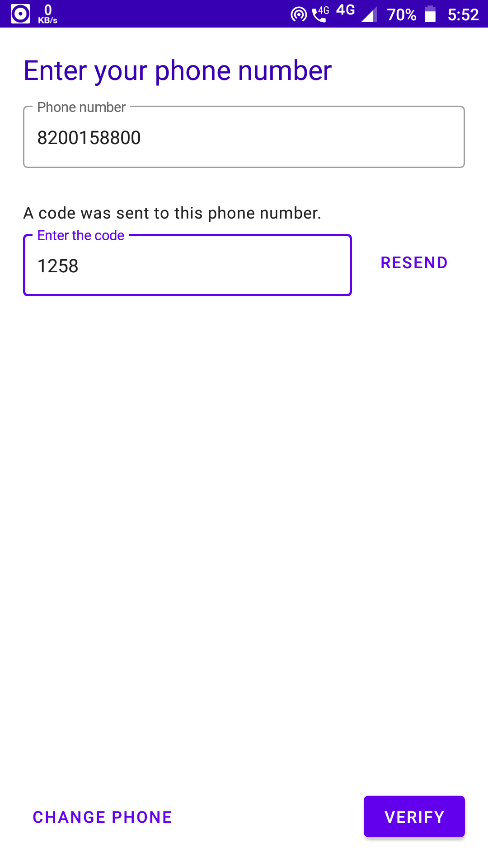
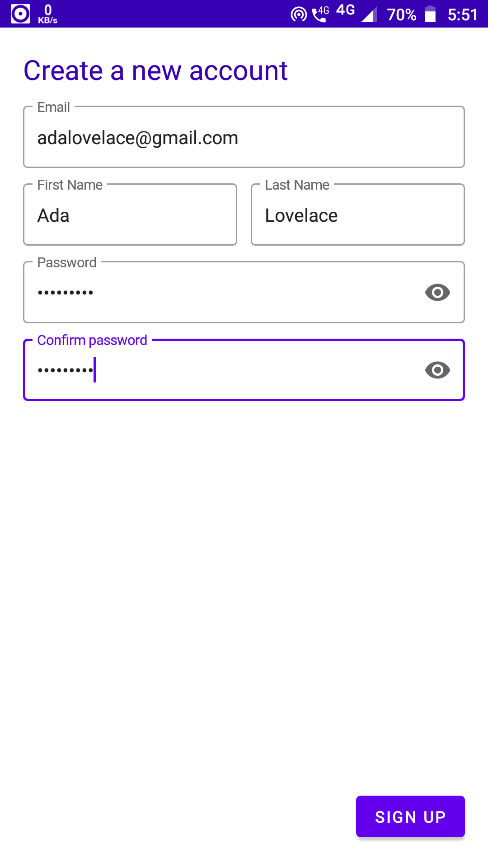
Main Application

Add volunteer

Special Permissions

Volunteers

Home

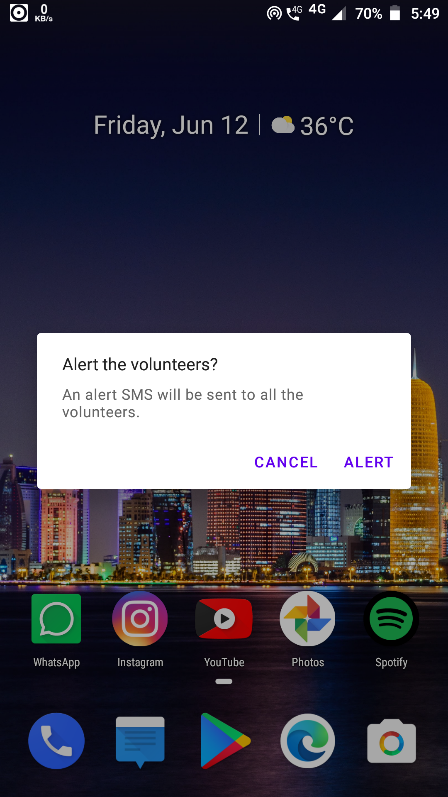
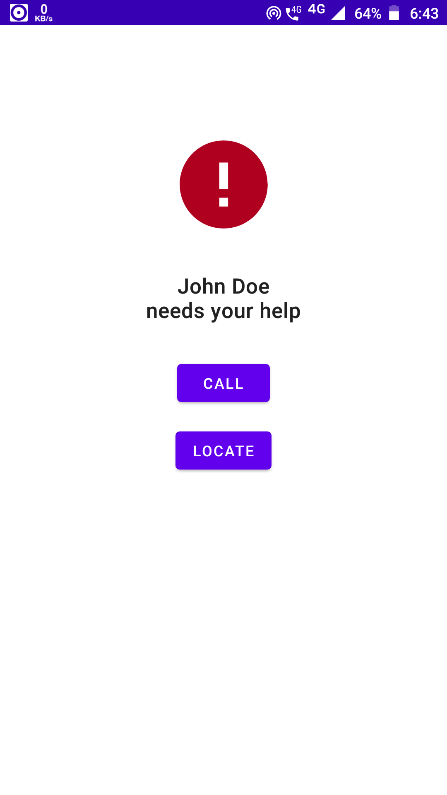
Authentication

Sign Up

Phone

Email

Password



Alert

Send Alert Dialog

# Chapter 6. Testing

Unit Testing

For unit testing, we will be using the **JUnit** testing framework along with the Android Testing Support Library, to access Android specific code inside these tests and also make use of **test doubles**.

*Trim phone*

It should remove the whitespaces and the country code (India - 91) from a mobile number, leaving only the essential digits.

|  |  |  |
| --- | --- | --- |
| **Test Case No.** | **When** | **Should** |
| 1 | Passed “+91 12345 67890” | Return “1234567890” |
| 2 | Passed “12345 67890” |
| 3 | Passed “911234567890” |

*Alert SMS Parser*

It should parse an SMS and extract the information related to a Victim from it, if it a valid Securify Alert.

|  |  |  |
| --- | --- | --- |
| **Test Case No.** | **When** | **Should** |
| 1 | SMS is of valid format | Extract information and return a Victim object |
| 2 | SMS is of invalid format | Return NULL |

*Alert SMS Builder*

It should build an SMS from a Victim object with a proper format.

|  |  |  |
| --- | --- | --- |
| **Test Case No.** | **When** | **Should** |
| 1 | Passed a Victim object | Return an SMS built from the Victim object with proper formatting |

UI Testing

For testing the user interface, we will use the **Espresso** testing framework provided by Android. It will test about the various changes in the UI in response to user’s interaction and other events.

*Home*

|  |  |  |
| --- | --- | --- |
| **Test Case No.** | **When** | **Should** |
| 1 | Clicked the Instant Alert switch | * Update the UI accordingly. * Update the shared preferences. |
| 2 | Clicked the “Send Alert” FAB | Show the dialog for sending the alert |

*Volunteers*

|  |  |  |
| --- | --- | --- |
| **Test Case No.** | **When** | **Should** |
| 1 | The screen is loaded | Display the list of volunteers |
| 2 | Clicked the “Add volunteer” FAB | * Show a Snackbar message if already 5 volunteers are added * If not, go to “Add volunteer” screen |
| 3 | Clicked the “Remove all” menu item | Show a dialog for confirmation of removal and update the UI accordingly |

*Add volunteer*

|  |  |  |
| --- | --- | --- |
| **Test Case No.** | **When** | **Should** |
| 1 | Clicked the “Contacts” icon | Go to Contacts app to select a contact |
| 2 | The screen is loaded after contact selection | Update the form fields with the selected contact’s details |
| 3 | Clicked the “Save” menu item | Save the volunteer in database if form input is valid |

# Chapter 7. Conclusion

We learnt how to create a mobile application using Android technology and by following some of the best development and design practices. Of course, we weren’t successful in developing each and every feature right away. We implemented an agile approach in which each feature is directly affected by the feasibility of various Android versions.

# Future Work

* Create more robust alerting using email and other notifications
* Continuous location update

# Bibliography

|  |  |
| --- | --- |
| [1] | “Android version history,” [Online]. Available: https://en.wikipedia.org/wiki/Android\_version\_history. |
| [2] | “Application Fundamentals | Android Developers,” [Online]. Available: https://developer.android.com/guide/components/fundamentals#Components. |
| [3] | “Applicaton Fundamentals | Android Developers,” [Online]. Available: https://developer.android.com/guide/components/fundamentals#Resources. |
| [4] | “Manifesto for Agile Software Development,” [Online]. Available: https://agilemanifesto.org/. |
| [5] | “What is Scrum?,” [Online]. Available: https://www.scrum.org/resources/what-is-scrum. |
| [6] | “Supported data types | Firebase,” [Online]. Available: https://firebase.google.com/docs/firestore/manage-data/data-types. |
| [7] | “Scrum Framework,” [Online]. Available: https://s3.amazonaws.com/scrumorg-website-prod/drupal/2016-06/ScrumFramework\_17x11.pdf. |