A Project Report

On

**“SIMPLE ANDROID SOS APPLICTION”**

*For the partial fulfillment of the requirement of 5th semester of*

Masters of Computer Application

of

Jorhat Engineering College

Affiliated To Dibrugarh University, Assam



Submitted By

**Trinayan Sankar Das**

MCA 5th Semester

Roll - MCA 18/16

**Zyandeep Baruah**

MCA 5th Semester

Roll - MCA 03/16

Under the guidance and supervision of:

**Mr. Abhinash Borah**

Assistant Professor

Department of MCA

Jorhat Engineering College

**JORHAT ENGINEERING COLLEGE**



**CERTIFICATE**

This is to certify that the project titled **“SIMPLE ANDROID SOS APPLICATION”** is a bona fide work, carried out by Trinayan Sankar Das and Zyandeep Baruah of MCA 5th semester for the partial fulfillment of the requirement for the Master of Computer Application.

I wish both of them all success in future.

**Mr. Joydip Sarmah**

HOD, Dept. of Computer Application

Jorhat Engineering College

**JORHAT ENGINEERING COLLEGE**



**CERTIFICATE**

This is to certify that the project titled **“SIMPLE ANDROID SOS APPLICATION ”** is a bonafide work, carried out by Trinayan Sankar Das and Zyandeep Baruah of MCA 5th semester for the partial fulfillment of the requirement for the Master of Computer Application.

I wish both of them all success in future.

**Mr. DhrubajyotiBaruah**

Assistant Professor,

Dept. of Computer Application

Jorhat Engineering College

**JORHAT ENGINEERING COLLEGE**



**CERTIFICATE**

This is to certify that the project titled “**SIMPLE ANDROID SOS APPLICATION ”** is a bona fide work, carried out by Trinayan Sankar Das and Zyandeep Baruah of MCA 5th semester for the partial fulfillment of the requirement for the Master of Computer Application.

I wish both of them all success in future.

**Mr. Abhinash Borah**

Assistant Professor,

Dept. of Computer Application

Jorhat Engineering College

**ACKNOWLEDGEMENT**

We take this opportunity to express our sincere thanks and gratitude to our project guide **Mr. Dhrubajyoti Baruah and Mr. Abhinash Borah** who motivated, encouraged and helped us to ensure that this project, **“SIMPLE ANDROID SOS APPLICATION ”** reached the stage of completion.

We also express our honest salutation and gratitude to **Mr. Joydip Sarmah**, HOD, Dept Of Computer Application, Jorhat Engineering College for the help and cooperation for successful completion of the project.

With due regards.

Your's Sincerely,

**Trinayan Sankar Das**

MCA 5th Semester

Roll - MCA 18/16

**Zyandeep Baruah**

MCA 5th Semester

Roll - MCA 03/16

**DECLARATION**

We hereby declare that the project entitled **“SIMPLE ANDROID SOS APPLICATION”** has been carried out by us and the contents have been submitted to the Department of Computer Application, for the partial fulfillment of the 5th semester of Masters of Computer Applications.

**Trinayan Sankar Das**

MCA 5th Semester

Roll - MCA 18/16

**Zyandeep Baruah**

MCA 5th Semester

Roll - MCA 03/16

**PROJECT OUTLINE**

* Title of the Project: “SIMPLE ANDROID SOS APPLICATION”
* Aim of the Project: The aim of this Android SOS application is to let a group of people know that a person is in a dangerous situation so that those people can rescue the person in danger.
* Purpose: 5th Semester Minor Project.
* Duration of the Project: 4 Months Approximately
* Project Done By: Trinayan Sankar Das (MCA-18/16) Zyandeep Baruah (MCA-03/16)

Guided By:

**Mr. Abhinash Borah**

Assistant Professor

Department of MCA

Jorhat Engineering College

**CHAPTER 1**

**INTRODUCTION**

**1.1 INTRODUCTION**

Personal safety applications or SOS applications aim to improve people’s safety while they are travelling and sense something amiss. As the harsh incidents around us increase in numbers, we are compelled to come up with solutions to ensure our safety.

SOS applications are used by women and solo travellers as these are the people at the highest risks in today’s society. With technologies such as video, alerts, alarms, GPS, SMS, etc., users in distress are able to alert their close friends and family about their situation.

Apparently, most of the SOS applications are made available by developers as freeware software applications, while some others are classified as “freemium” applications, where the general features are available for free and advanced ones need users to go for premium subscriptions at reasonable prices. The android application **“bSafe”** that we have developed as part of our 5th semester project is a free android application for users.

**1.2 PROBLEM DEFINATION**

As much as we would like to get rid of them, panic or emergency situations are unavoidable and usually unexpected. The nature and consequences of these situations can vary significantly and in worst cases also be life threatening. Therefore, it would be really nice to have some mechanism by which we can notify certain people about such circumstances and increase the chances of receiving help as soon as possible. The need for such a mechanism increases even more as in this era of technology, platforms exist to support them. One such platform and a very common one in that is a Smartphone. Almost everyone today carry a Smartphone with them as they become more and more affordable and easily available. Also within the Smartphone market Android is the clear leader in terms of market share. According to one report, 78.1 % of the total Smart phones sold in 2013 were the Smart phones that run-on Android Operating System. Hence developing an Android application becomes an obvious choice.

**1.3 THE PROPOSED SYSTEM & ITS OBJECTIVES**

The proposed system is an android application and the aim of which is **to let a group of people know that a person is in a dangerous situation so that those people can rescue the person in danger.**

The following are main objectives of our proposed system:

1. This android app will inform those people by sending an SMS containing a piece of text, which may be default or custom, along with the user’s current location in the form of a URL link which will open up in Google Map.
2. The SOS SMS can be triggered in the following three ways...
   1. By pressing the POWER BUTTON 4 times continuously.
   2. From the App Widget present on the home screen.
   3. And from the within the app itself.
3. The SOS app provides the user to add maximum three emergency contacts numbers form his/hers contact list. And the app will send SOS SMS to only those contacts. The user can also change the emergency contacts.
4. The app has a default SOS text such as “I AM IN DANGER”, but user can also add his/her own custom text.
5. When the SOS SMS is triggered by pressing the POWER BUTTON, the app will inform the user that SMS has been sent out successfully by providing a haptic feedback (means vibrating the phone for 1000 ms).

**CHAPTER 2**

**FEASIBILITY STUDY**

**2.1 INTODUCTION**

The main aim of the feasibility study activity is to determine whether it would be financially and technically feasible to develop the product. The feasibility study activity involves the analysis of the problem and collection of all relevant information relating to the product such as the different data items which would be input to the system, the processing required to be carried out on these data, the output data required to be produced by the system as well as various constraints on the behavior of the system.

There are three key considerations involved in the feasibility analysis. They are:

* Economic Feasibility
* Technical Feasibility
* Behavioral Feasibility

**2.2 ECONOMIC FEASIBILITY**

In economic Feasibility, which is also known as cost benefit analysis, benefit expected from the candidate system is compared with cost. If the benefit out weights cost, then the decision is made to design and implement the system. Otherwise, further justification or alteration in the proposed system will have to be made.

**2.3 TECHNICAL FEASIBILTY**

Technical Feasibility centers around the existing system (hardware, software etc.) and to what extend it can support the proposed system. The study here involves financial consideration too, in order to accommodate technical enhancement. Technically, the development of the system is feasible since all the resources required for the development of the proposed system are easily available.

.

**2.4 BEHAVIORAL FEASIBILITY**

In general, people resistant to change and smart phones have been known to facilitate change. A survey should be made of how strong a reaction the user staff is likely to have towards the development of mobile applications.

As more and more people are purchasing smart phones, the attitude towards the development of mobile applications seems to be positive. Moreover an easy, unambiguous, attractive GUI will attract more and more people towards the applications that are being built.

**2.5 CONCLUSION**

From the observation made in the feasibility study described above, it was recommended that proposed system is completely feasible for its development and implementation.

CHAPTER 3

REQUIREMENT ANALYSIS

**3.1 REQUIREMENT GATHERING**

Requirements gathering are one of the most important phase of a software development life cycle. It is the phase that tells us what is the system supposed to do and drives the other phases in the life cycle. Requirement gathering for the SOS app started with brain storming and discussion with other students as to what features are the most essential in a panic situation. This led to the most basic and initial draft of requirements for the application. Requirements were also collected by looking at other personal safety applications that are out there. A brief study of the functionality of the applications helped me to refine and narrow down the requirements even further. One important thing to learn for these applications was the simplicity of their design. This helped me to design an effective and simple UI design for my application. The next step for requirements understanding was to look for existing solutions and similar applications in the Android market. A careful study of these applications, adding other important features and removing unnecessary features was done.

We discussed with our project guide regularly and he helped me to refine the requirements and user interface even further to set a clear set of functional requirements for the application.

The major functional requirements for the SOS app are –

1. The user of the application should be ask to add maximum three emergency contacts numbers form his/hers contact list. And the app should send SOS SMS to only those contacts. The user can also change the emergency contacts.
2. The SOS SMS should be triggered in the following three ways...
   1. By pressing the POWER BUTTON 4 times continuously.
   2. From the App Widget present on the home screen.
   3. And from the within the app itself.
3. The app should have a default SOS text such as “I AM IN DANGER”, but user can also add his/her own custom text.
4. When the SOS SMS is triggered by pressing the POWER BUTTON, the app should inform the user that SMS has been sent out successfully by providing a haptic feedback (means vibrating the phone for 1000 ms).
5. This android app should send the SOS message as an SMS that must contain a piece of text, which may be default or custom, along with the user’s current location in the form of a URL link which should open up in Google Map.

Other non-functional requirements for the application are –

1. Providing a simple and elegant UI for the main screen. This is necessary as the user would usually come on to this screen in case of a panic or emergency and hence each button should be clearly visible and easily pressed.
2. The application should be able to display a informative dialogue box in case the phone’s GPS is turned off. Even if the user does not turned on the GPS the app should gather the user’s current location using cell towers.

**3.2 Requirement Specifications**

**Software Requirements :**

These requirements are separated based on whether we are developing the app or running the app on a device.

**For development:**

Operating System: Windows 7 or higher /Linux

IDE: Android Studio

Technologies used: Java, XML, Android, Fused Location Provider

Android Emulator: Genymotion, Android Studio AVD

**For running on a device:**

Operating System: Android 5.0 or higher

Cellular capabilities for SMS messages and GPS.

**Hardware Requirements :**

**For development:**

Processor: Intel i3 6th Gen or Higher

RAM: 4GB or Higher

Space on disk: 4 GB (at the least)

**For running on a device:**

Device: Phone or tablet running Android 5.0 or higher

Disk space: 2 GB(at the least)

**CHAPTER 4**

**TECHNOLOGY OVERVIEW**

**4.1 Android**

Android is one of the most widely used Mobile Operating System today. It is a software bunch comprising not only of the Operating System but also middleware and key applications. Some of the most important features of an Android operating system is that it enables reuse and replacement of components, it is optimized for mobile devices and tablets, it is based on the open source Web kit engine and supports 2-D and 3-D graphics using OpenGL-ES standard.

**Android Architecture :**

The Android operating system is implemented as a stack of different layers of software. The following image depicts these different layers:

 **Figure: Android Architecture**

**Linux Kernel –** This is the layer at the very bottom of the Android architecture. All other layers run on top of the Linux kernel and rely on this kernel to interact with the hardware. This layer contains all the essential hardware drivers which help to control and communicate with the hardware. It provides the basic functionality like Process Management, Memory Management and Device Management like Camera, Display, Flash etc.

**Libraries –** This is a set of common functions of the application framework that enables the device to handle different types of data. Some of the most important set of libraries that are included are – Web kit which is the browser engine to display HTML, OpenGL used to render 2-D or 3-D graphics on to the screen, SQLite which is a useful repository for storing and sharing of application data.

**Android Runtime –** The Android runtime mainly consist of the Dalvik Virtual Machine (DVM)**.**DVM is very much like the standard Java Virtual Machine (JVM) except that it is optimized formobile devices that have low processing power and low memory. DVM generates a .dex filefrom the .class file at compile time and provides higher efficiency in low resources devices. Eachapplication has its own process and an instance of DVM. Android runtime also provides corelibraries that enable the Android developers to create applications using the Java language.

**Application Framework-** These are some standard class files that are available to the developer for use. An application can directly interact with them and make use of them. The application framework provides the most basic functionality of the phone like Location Manager, Content Providers etc.

**Applications –** This is the topmost layer in the architecture and the layer where the application that we develop fits in. This layer provides several pre-installed applications that are default for certain things like Contacts Books, Browser etc.

**4.2 Android Studio**

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA . On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

* A flexible Gradle-based build system
* A fast and feature-rich emulator
* A unified environment where you can develop for all Android devices
* Instant Run to push changes to your running app without building a new APK
* Code templates and GitHub integration to help you build common app features and import sample code
* Extensive testing tools and frameworks
* Lint tools to catch performance, usability, version compatibility, and other problems
* C++ and NDK support
* Built-in support for [Google Cloud Platform](https://cloud.google.com/tools/android-studio/docs/), making it easy to integrate Google Cloud Messaging and App Engine

**Project Structure:**

Each project in Android Studio contains one or more modules with source code files and resource files. Types of modules include:

* Android app modules
* Library modules
* Google App Engine modules
  + By default, Android Studio displays your project files in the Android project view, as shown in figure 1. This view is organized by modules to provide quick access to your project's key source files.
  + All the build files are visible at the top level under **Gradle Scripts** and each app module contains the following folders:
* **manifests**: Contains the AndroidManifest.xml file.
* **java**: Contains the Java source code files, including JUnit test code.
* **res**: Contains all non-code resources, such as XML layouts, UI strings, and bitmap images.

The Android project structure on disk differs from this flattened representation. To see the actual file structure of the project,select**Project** from the **Project** dropdown (in figure 1, it's showing as **Android**).

You can also customize the view of the project files to focus on specific aspects of your app development. For example, selecting the **Problems** view of your project displays links to the source files containing any recognized coding and syntax errors, such as a missing XML element closing tag in a layout file.



**Figure1:** The project files in Android view.



**Figure 2:**The project files in Problems view, showing a layout file with a problem.

**The User Interface:**

The Android Studio main window is made up of sever.

**Figure 3:**The Android Studio main window.

1. The **toolbar** lets you carry out a wide range of actions, including running your app and launching Android tools.
2. The **navigation bar** helps you navigate through your project and open files for editing. It provides a more compact view of the structure visible in the **Project** window.
3. The **editor window** is where you create and modify code. Depending on the current file type, the editor can change. For example, when viewing a layout file, the editor displays the Layout Editor.
4. The **tool window bar** runs around the outside of the IDE window and contains the buttons that allow you to expand or collapse individual tool windows.
5. The **tool windows** give you access to specific tasks like project management, search, version control, and more. You can expand them and collapse them.
6. The **status bar** displays the status of your project and the IDE itself, as well as any warnings or messages.

**4.3 JAVA**

Java programming language was originally developed by Sun Microsystems which was initiated by James Gosling and released in 1995 as core component of Sun Microsystems' Java platform (Java 1.0 [J2SE]).

The latest release of the Java Standard Edition is Java SE 8. With the advancement of Java and its widespread popularity, multiple configurations were built to suit various types of platforms. For example: J2EE for Enterprise Applications, J2ME for Mobile Applications.

The new J2 versions were renamed as Java SE, Java EE, and Java ME respectively. Java is guaranteed to be **Write Once, Run Anywhere.**

Java is −

* **Object Oriented** − In Java, everything is an Object. Java can be easily extended since it is based on the Object model.
* **Platform Independent** − Unlike many other programming languages including C and C++, when Java is compiled, it is not compiled into platform specific machine, rather into platform independent byte code. This byte code is distributed over the web and interpreted by the Virtual Machine (JVM) on whichever platform it is being run on.
* **Simple** − Java is designed to be easy to learn. If you understand the basic concept of OOP Java, it would be easy to master.
* **Secure** − With Java's secure feature it enables to develop virus-free, tamper-free systems. Authentication techniques are based on public-key encryption.
* **Architecture-neutral** − Java compiler generates an architecture-neutral object file format, which makes the compiled code executable on many processors, with the presence of Java runtime system.
* **Portable** − Being architecture-neutral and having no implementation dependent aspects of the specification makes Java portable. Compiler in Java is written in ANSI C with a clean portability boundary, which is a POSIX subset.
* **Robust** − Java makes an effort to eliminate error prone situations by emphasizing mainly on compile time error checking and runtime checking.
* **Multithreaded** − With Java's multithreaded feature it is possible to write programs that can perform many tasks simultaneously. This design feature allows the developers to construct interactive applications that can run smoothly.
* **Interpreted** − Java byte code is translated on the fly to native machine instructions and is not stored anywhere. The development process is more rapid and analytical since the linking is an incremental and light-weight process.
* **High Performance** − With the use of Just-In-Time compilers, Java enables high performance.

**4.4 XML:**

XML stands for **E**xtensible **M**arkup **L**anguage. It is a text-based markup language derived from Standard Generalized Markup Language (SGML).

XML tags identify the data and are used to store and organize the data, rather than specifying how to display it like HTML tags, which are used to display the data. XML is not going to replace HTML in the near future, but it introduces new possibilities by adopting many successful features of HTML.

There are three important characteristics of XML that make it useful in a variety of systems and solutions −

* **XML is extensible** − XML allows you to create your own self-descriptive tags, or language, that suits your application.
* **XML carries the data, does not present it** − XML allows you to store the data irrespective of how it will be presented.
* **XML is a public standard** − XML was developed by an organization called the World Wide Web Consortium (W3C) and is available as an open standard.

**XML Usage:**

A short list of XML usage says it all −

* XML can work behind the scene to simplify the creation of HTML documents for large web sites.
* XML can be used to exchange the information between organizations and systems.
* XML can be used for offloading and reloading of databases.
* XML can be used to store and arrange the data, which can customize your data handling needs.
* XML can easily be merged with style sheets to create almost any desired output.
* Virtually, any type of data can be expressed as an XML document.

**CHAPTER 5**

**SYSTEM DESIGN**

**5.1 Design Diagrams**

**Activity Diagram:**

We use **Activity Diagrams** to illustrate the flow of control in a system and refer to the steps involved in the execution of a use case. We model sequential and concurrent activities using activity diagrams. So, we basically depict workflows visually using an activity diagram. An activity diagram focuses on condition of flow and the sequence in which it happens. We describe or depict what causes a particular event using an activity diagram.

UML models basically three types of diagrams, namely, structure diagrams, interaction diagrams, and behavior diagrams. An activity diagram is a **behavioral diagram** i.e. it depicts the behavior of a system.

An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed. We can depict both sequential processing and concurrent processing of activities using an activity diagram. They are used in business and process modelling where their primary use is to depict the dynamic aspects of a system.

The various components used in the diagram and the standard notations are explained below.

### Activity Diagram Notations –

1. **Initial State –** The starting state before an activity takes place is depicted using the initial state.

UML-State-Diagram

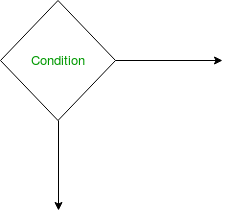
1. **Action or Activity State –** An activity represents execution of an action on objects or by objects. We represent an activity using a rectangle with rounded corners. Basically any action or event that takes place is represented using an activity.

UML-Activity-Diagram

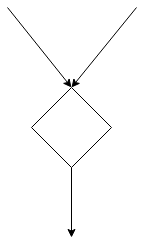
**3. Action Flow or Control flows –** Action flows or Control flows are also referred to as paths and edges. They are used to show the transition from one activity state to another.

UML-Object-Diagram

**4.Decision node and Branching –** When we need to make a decision before deciding the flow of control, we use the decision node.



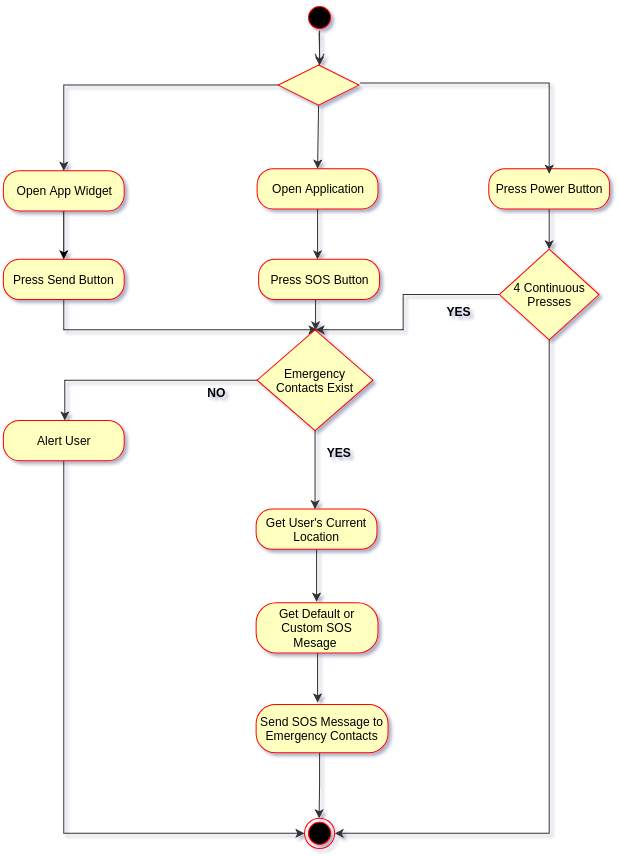
**5.Merge or Merge Event –**Scenarios arise when activities which are not being executed concurrently have to be merged. We use the merge notation for such scenarios. We can merge two or more activities into one if the control proceeds onto the next activity irrespective of the path chosen.



**6.Final State or End State –** The state which the system reaches when a particular process or activity ends is known as a Final State or End State. We use a filled circle within a circle notation to represent the final state in a state machine diagram. A system or a process can have multiple final states.

UML-State-Diagram

**ACTIVITY DIAGRAM OF THE SOS APPLICATION**



**CHAPTER 6**

**IMPLEMENTATION**

The SOS app has one main screen and three sub screens. One of which lets the user to add and delete his/hers emergency contacts. And the other one is Settings screen for setting or resetting the SOS text. The app has an Info Screen as well for guiding the user how to use the application.

Below are the screenshots of the app during development:

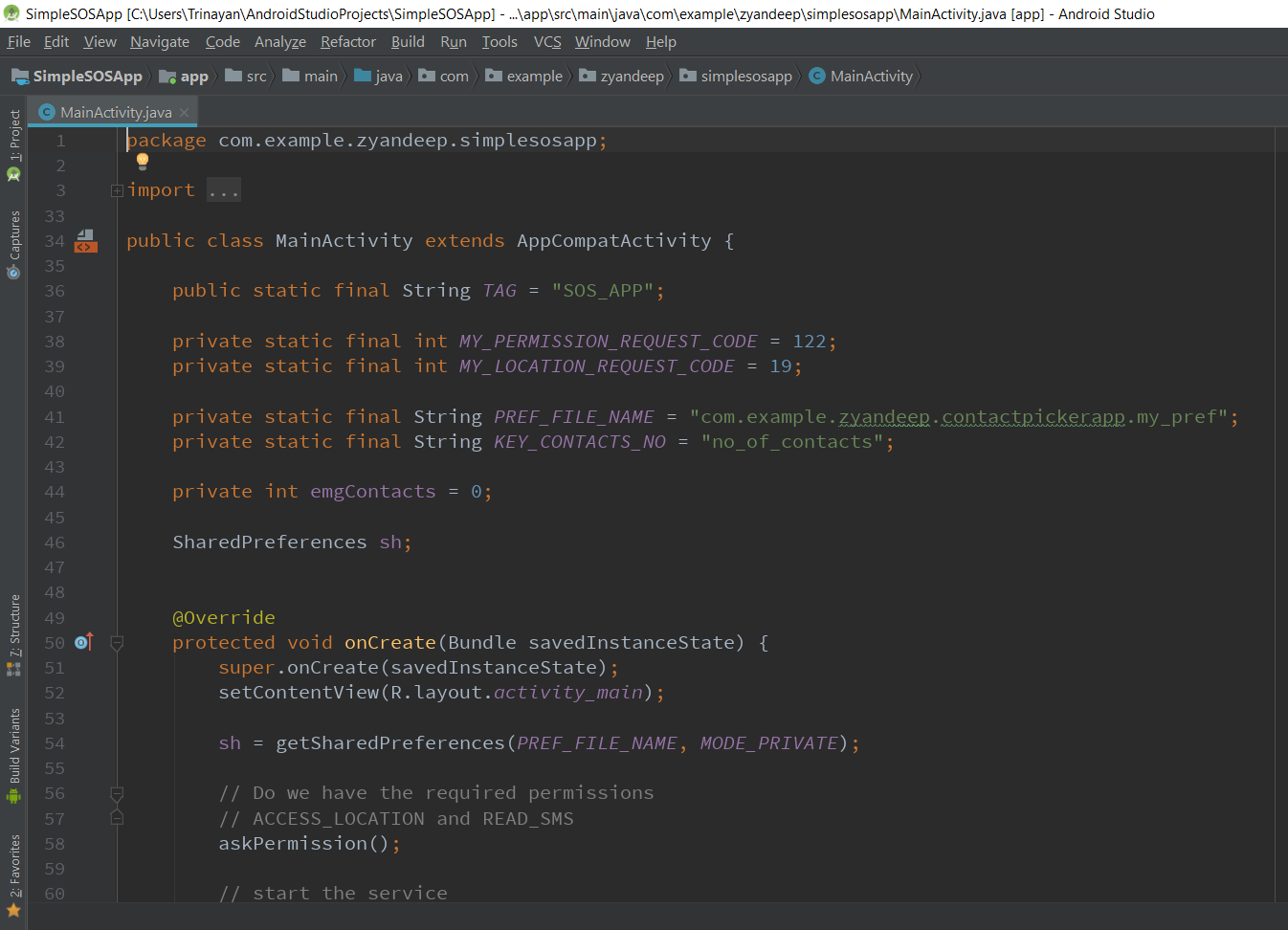


Fig: The MainActivity of the app

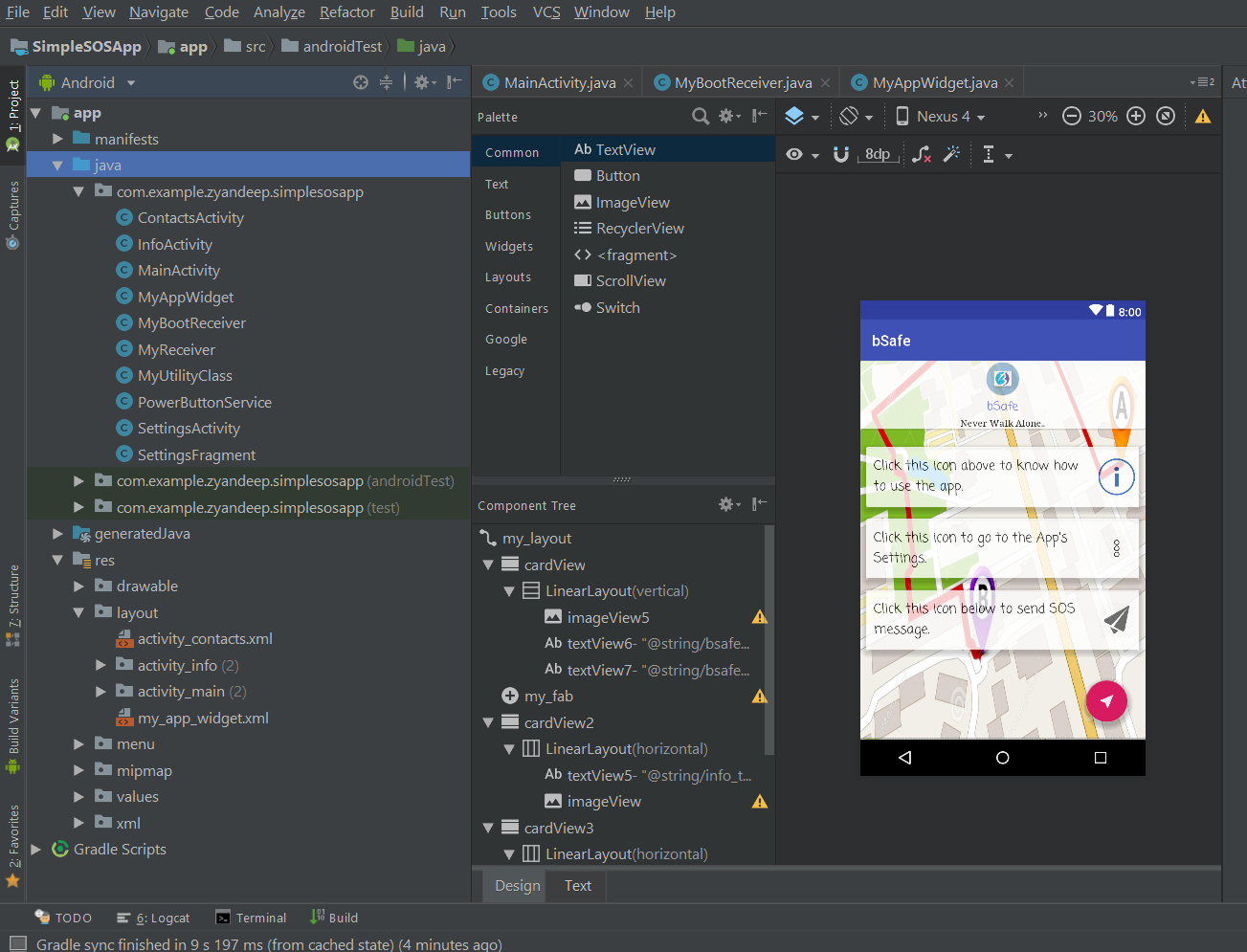
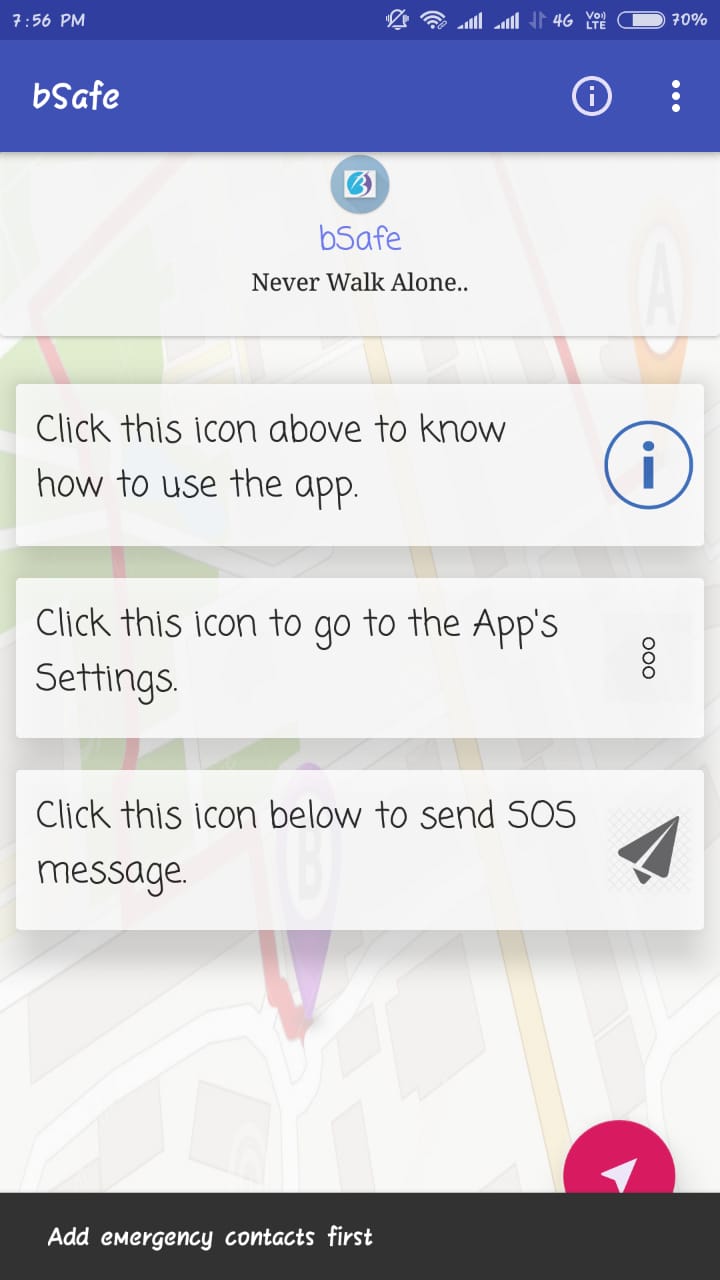
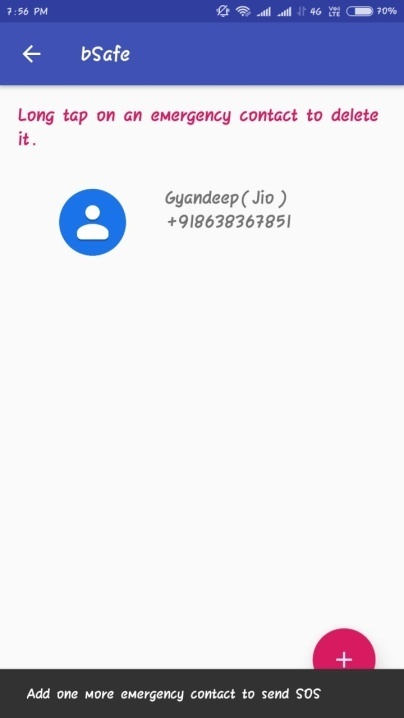


Fig: The preview of the XML layout of the Main Screen

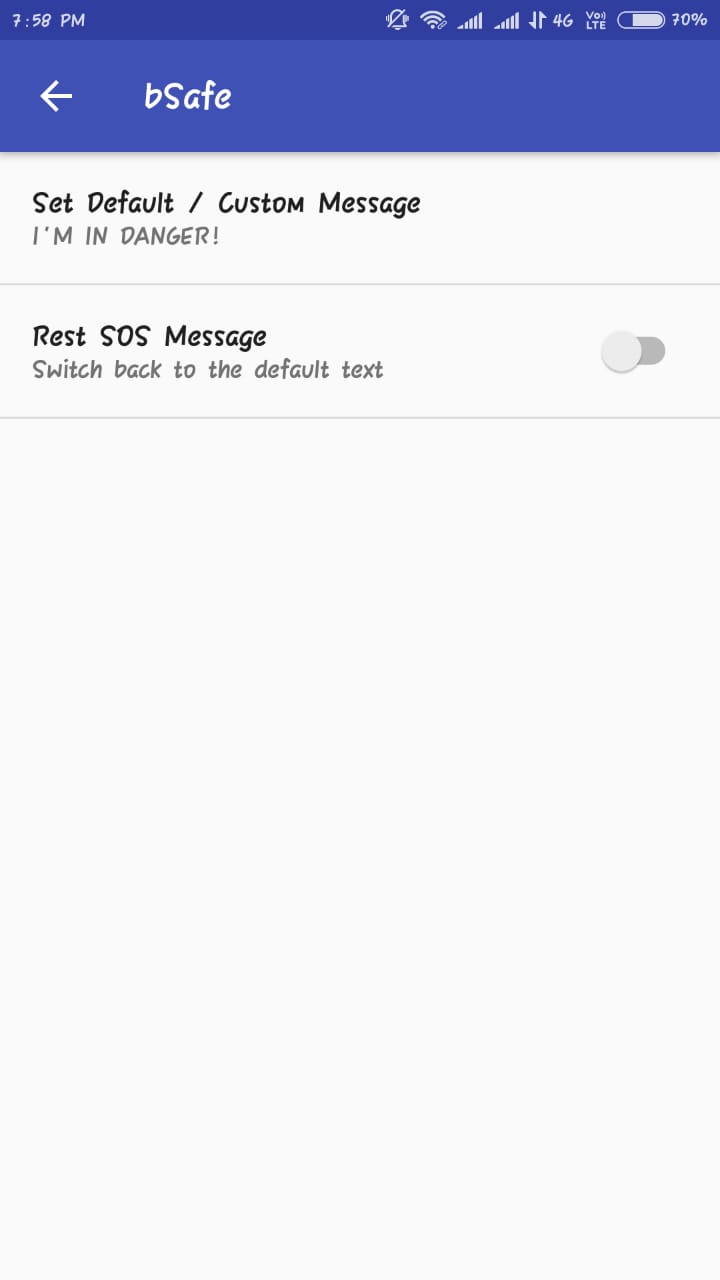
**Some Screenshots of “bSafe”:**



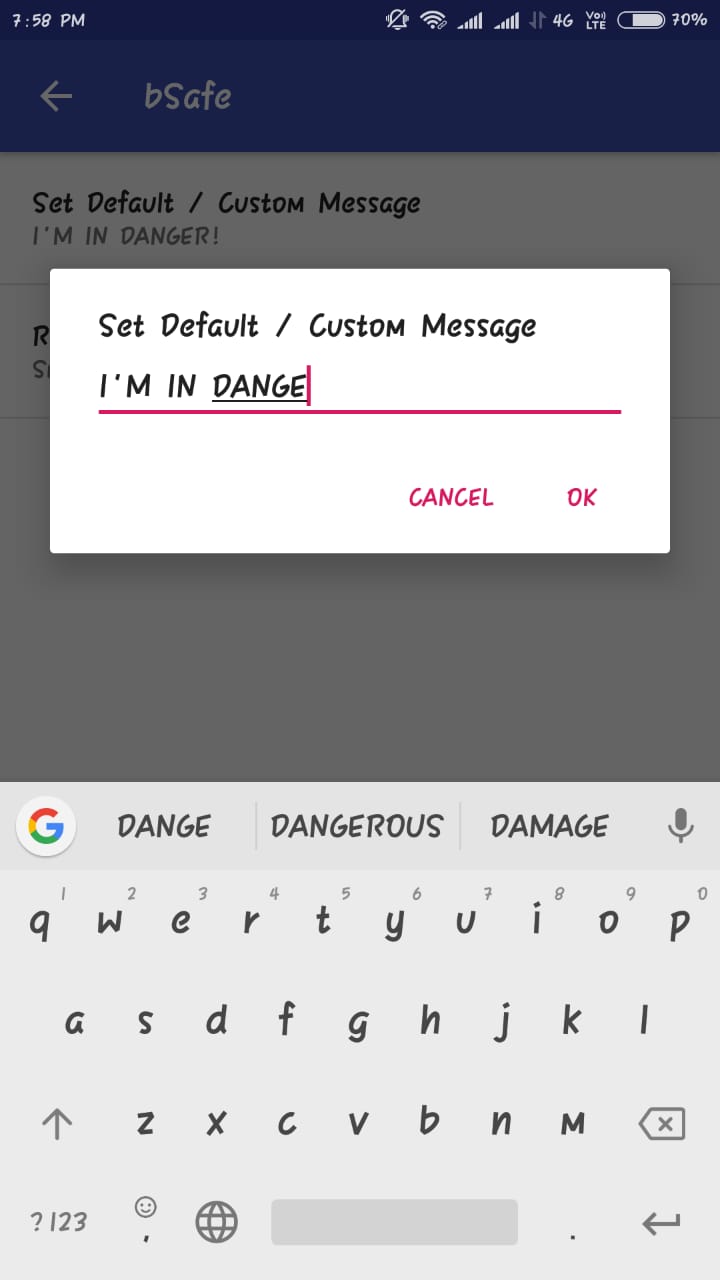
Home Screen



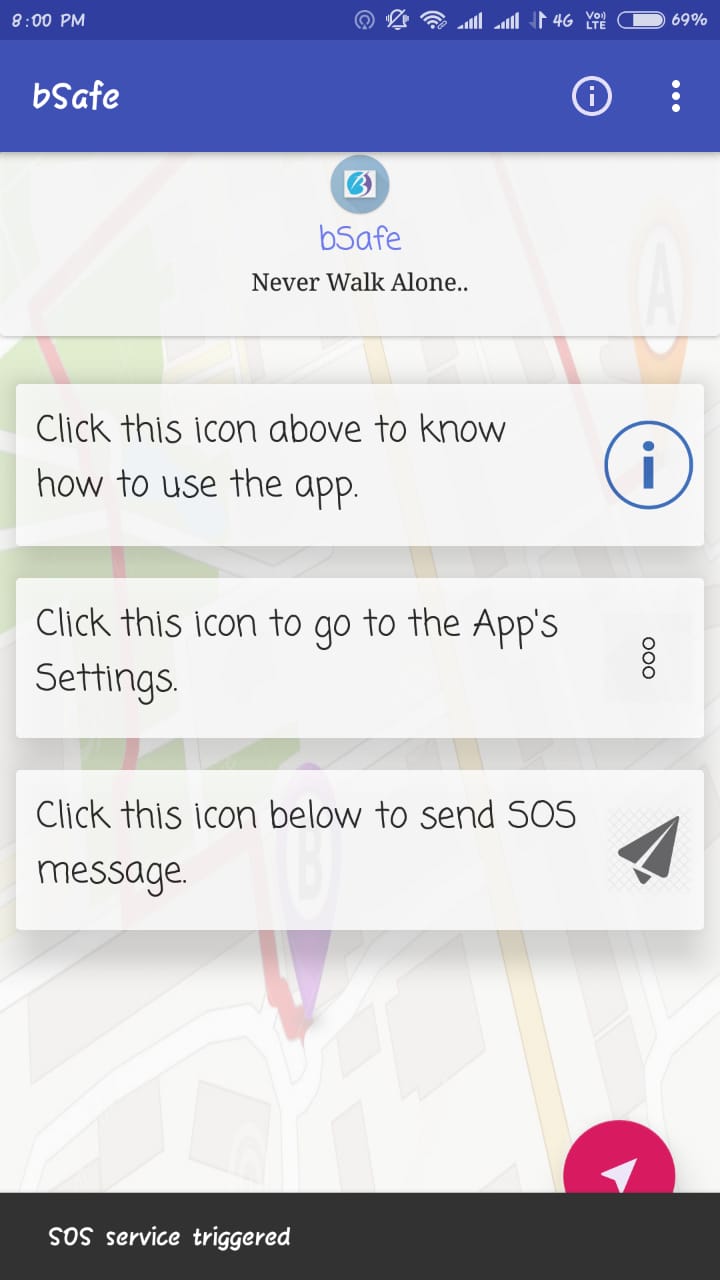
Emergency Contact Screen



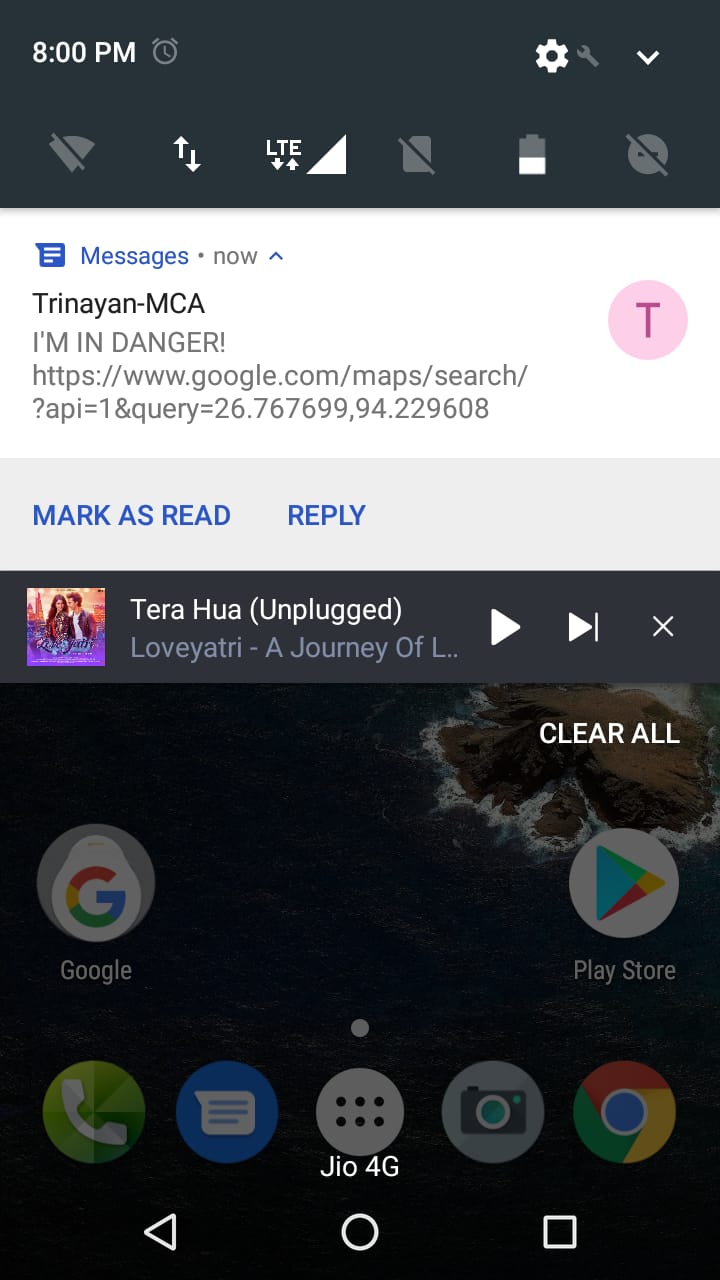
SOS Message Setting Screen I



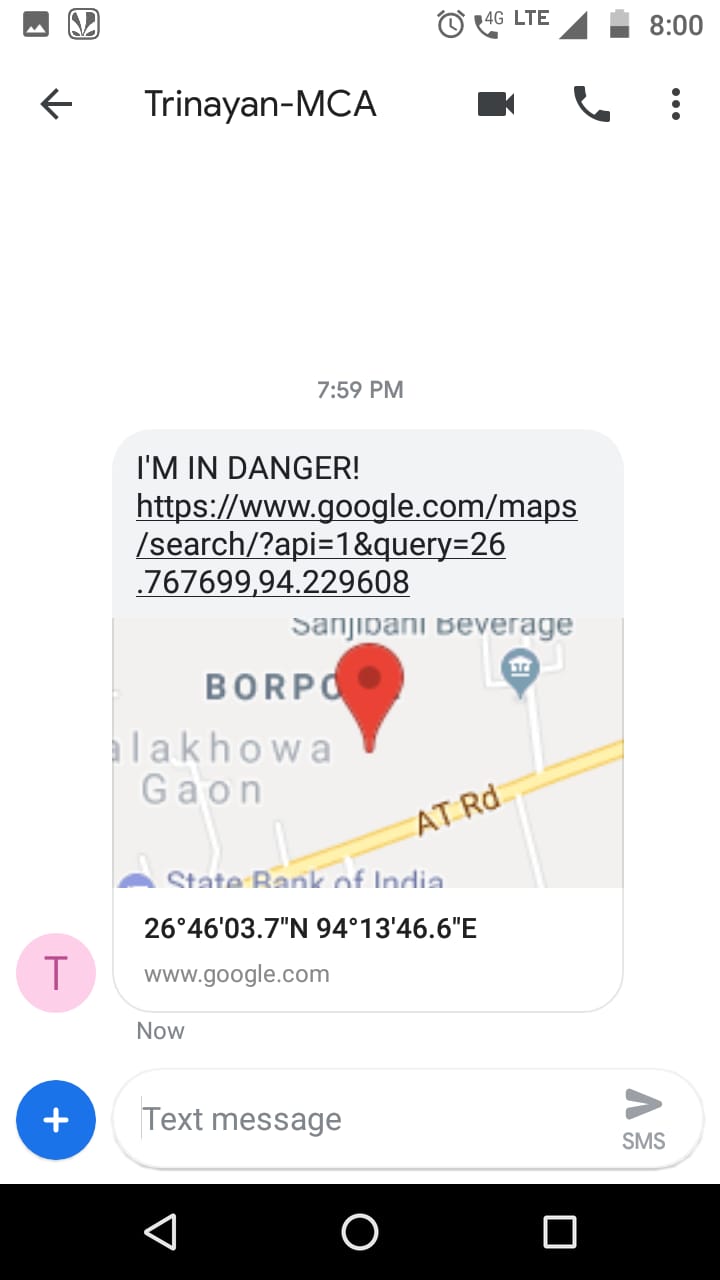
SOS Message Setting Screen II



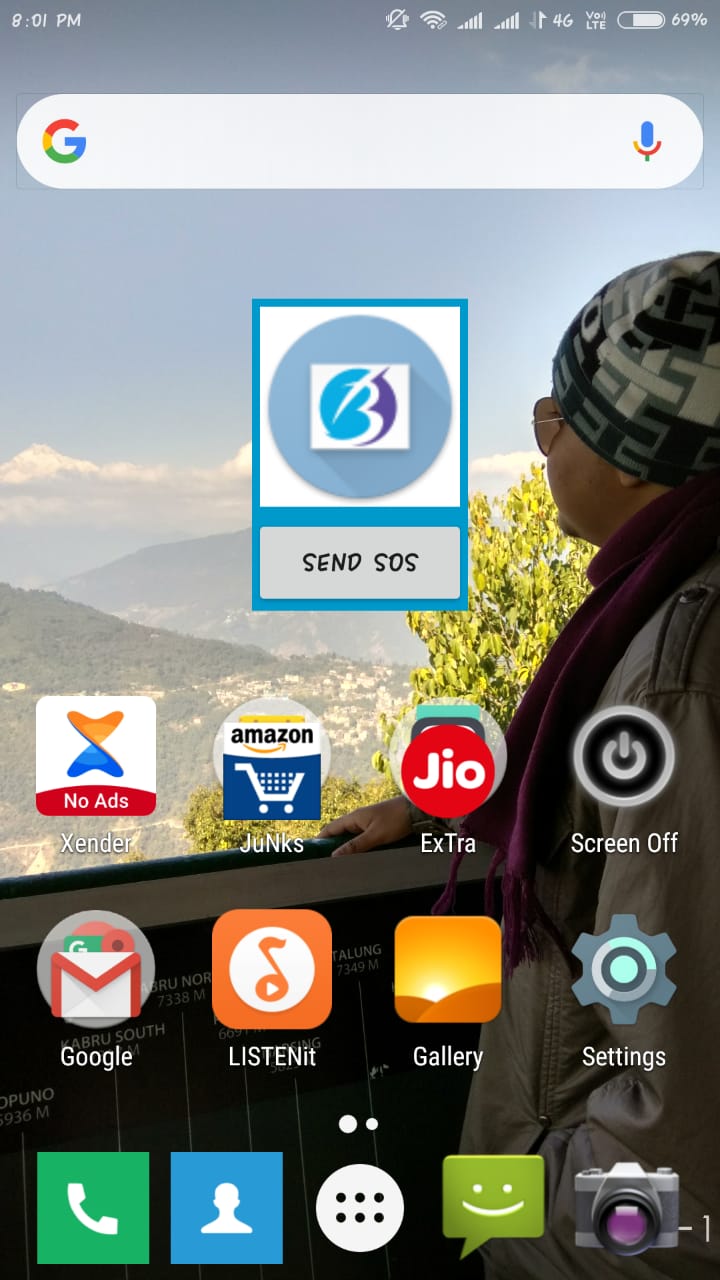
Snap of MainActivity when SOS service triggered.



Snap of SOS message received by one of the Emergency Contact



Snap of SOS SMS that contains user’s location in the form of Google Map URL



bSafe Widget

**CHAPTER 7**

**FUTURE WORK**

The current work on the SOS app has a lot of essential features that would be used in case of an emergency situation like sending text messages with user’s current location within the app on tap of a single button, activating the SOS service by pressing phone’s power button, or by using an app Widget. An app for such a purpose has a lot of scope for enhancement. In the future, the app may include features like –

1. This Emergency app will also provide the current country emergency numbers - Police, Fire, Ambulance so that users can ask for help directly within the app in case of such emergency.
2. The app can also listen to incoming messages from the set contacts. If these messages have a pre-defined text like “TRACK LOCATION” the app can reply with a text message containing the current location periodically, say after 3 minutes. This can be very helpful because this way the person in danger can be tracked constantly.
3. This Emergency app can come with "Photo mail" feature that can capture a photo and quickly send an email to your pre-filled email addresses containing the photo, the SOS message and the user’s location.

**CHAPTER 7**

**CONCLUSION**

SOS is an essential app to have on a Smartphone. It is a personal security app that lets you send notifications to certain people via text messages in case of emergencies on the tap of a single button. The app also gets the user’s current location to let a group of people know where the user is in case of an emergency situation. The SOS messages also have this location information.

SOS app was our very first attempt at an Android application. It gave us very good exposure to the Android platform and mobile development in general. The app enabled us in understanding the basic of Android development, learning about various APIs available for Android and Google’s Material Design for modern android app designing.

**CHAPTER 8**

**BIBLIOGRAPGY**

1. **Google’s Developer Guides:**

<https://developer.android.com/guide/>

1. **Google Developer Training:** <https://developers.google.com/training/android/>
2. **YouTube: Google Developers India** <https://www.youtube.com/playlist?list=PLlyCyjh2pUe9wv-hU4my-Nen_SvXIzxGB/>
3. **Vogella Android Tutorial** <http://www.vogella.com/tutorials/android.html/>