Smartphone Instrumentation for Collapse Detection of Buildings

A Mobile Crowd-Sourcing Approach Towards Earthquake Detection and Early Warning System

|  |
| --- |
| APPLIED COMPUTING PROJECT I & II  SYSTEM IMPLEMENTATION DOCUMENT |

|  |  |
| --- | --- |
|  |  |
|  | |  | | --- | |  | |

|  |  |  |
| --- | --- | --- |
| ZEYUN ZHU  2382881  zeyunzhu@gmail.com | HAEJONG DONG  2292191  s2haejong@gmail.com | PERTTU PITKÄNEN  2261724  perttu.pitkanen@student.ouLu.fi |

# **Introduction**

## Background and Motivation

The most common information available immediately following a catastrophic event such as an earthquake is its magnitude and epicenter location. However, an accurate and a full extent of the damage assessment requires time and is currently mostly provided by estimates. For instance, the full extent of the damage from the 1995 Kobe earthquake, Japan, was not recognized by the central government in Tokyo until many hours later. This greatly affected the rescue and recovery operations. Products such as ‘ShakeMaps’ [10] come close to the rescue by providing near real-time estimates using the sensors that are available from traditional seismic networks such as the Southern California Seismic Network (SCSN). A trade-off associated with these hi-fidelity seismic sensors is its sparse distribution (approximately 10 km in case of SCSN [11]) resulting in maps of low-resolution. Increasing the density of such hi-fidelity networks beyond that is needed for its basic function of locating the earthquake epicenter is cost prohibitive. The cost factor has been partially addressed by the use of open-network of low-cost microelectromechanical systems (MEMS) sensors that are hosted by volunteers [11]. However, the count of actual collapse of buildings and fatality is not obvious from these methods.

Another popular approach towards obtaining earthquake measurements data is the use of crowdsourcing. The “Did You Feel It” (DYFI) product of the US Geological Survey does this with a simple post-earthquake questionnaire [12] . The form of the questionnaire and the method for assignment of intensities are based on an algorithm developed by Dengler and Dewey for determining a "Community Decimal Intensity" [13]. Recent mild but widely-felt earthquakes in Los Angeles region have produced over 40,000 entries; supporting the likelihood of the popularity of such systems. However, this system has its caveats. One drawback of this form of sensing is that the human responders in the areas of heavy shaking usually do not make the data entry their first priority, and hence information from the most critical areas is usually late. Further, as this method of sensing requires users’ attention during times of peril, they could be considered hazardous in nature.

# **Motivation**

As a consequence, earthquakes and specifically the building collapses that happen during the earthquake are a serious threat to many people living in high earthquake risk areas. Collapse of buildings is one of the primary causes of fatality in any earthquake. Although hi-fidelity seismic sensors stations are deployed for early detection of earthquake tremors, little effort has been put to detect collapse of buildings in real time. The post-earthquake period of detecting collapsed buildings mostly involve image processing of aerial photographs. This method introduces a significant delay between the time of occurrence of the actual ‘collapsing’ event and the time the Emergency Response Units (ERUs) are deployed, thus causing a rise in fatality rates.

In this project, we focus on sensing the earthquakes and detecting collapsed buildings utilizing the smartphones as the distributed sensors. It introduces a system to cater to the needs of three different user groups - smartphone users, seismologists, and emergency service responders with the help of AWARE framework.

## Scope and Objectives

Our system is built on the Aware platform. Our mobile plugin detects phone fall and provides location information of nearby building collapses. The plugin shall also provide information on safety and first aid. Here we propose the design of a two-tier mobile client-backend server architecture that will use a dense open crowd-sourced network of smartphones. The smartphones will act as distributed sensors that enable our system to monitor and detect collapse of buildings in real time whilst providing crucial location information. By using the in-built accelerometer and gyroscope sensors of the smartphone our algorithms will detect the free-fall of smartphones during an earthquake. Depending on the number of smartphones that fall concurrently, our server will determine whether an ‘actual’ collapse of building has occurred. This being a time critical system, special attention will be given to the synchronization of the individual device clocks.

## Major assumptions and constraints

Assumptions and constraints reagarding the plugin

An assumption upon the smartphones is that users leave their smartphones on the table for the most of time throughout the day. However, evaluation of our system will indicate the feasibility and effectiveness of our approach when the system is deployed in real life environment where users tend to carry their smartphones with them.

The plugin measures the acceleration along the three axes: x,y and z. We have decided that the plugin will trigger a fall event when vector sum of acceleration sqrt(x^2 +y^2+ z^2) is less than 0.3. The threshold has been acquired by analyzing the accelerometer values during various drops with contemporary smatphone. This threshold may not work on every phone, and the threshold might need to be adjusted depending on the phone used. If selcected threshold doesn’t match the real-life fall event’s accelerometer values the plugin could result in false positives or false negatives depending on wether the threshold is too high or too low.

Another strong assumption that the plugin software has, is that the phone would be in a free fall during an actual building collapse. In theory it would seem probable that phones among other objects would be in a free fall just enough to trigger the fall detection. Altough we haven’t been capable on testing system functionality in a realistic situation. It is unsure what kind of values the accelerometer will record during collapse and the values could depend on the location of the phone inside the building and wether the phone can fall freely or if encounters obstacles like other objects during the collapse and thus doesn’t enter a free fall. Multiple free falls could be also detected with phones during an earthquake without the building being collapsed. The phones could, for example, fall from tables at the same time in multiple locations inside the building.

One constraint that the phone has is that the floating point calculations needed to calculate the vector sum of the three accelerometer axis are very calculation-heavy operations. When the calculations are made several times in a secon it is very demanding on the smartphones processor. Increased processor usage consequently shortens the battery lifetime in smartphones.

**Assumptions and constraints regarding the server**

Another assumption upon the fall defining algorithm is that when a fall event is observed by the server it generates a circle with a radius of 125 meters at the location of the event. once this happenens the server waits for a pre-defined time(timewindow) of ‘60’ seconds and registers all the incoming clients as a group of fall event. we assume when there are more than two fall events in the same circle that there will be probability of a building collapsed in the center coordiate of the circle. The probability varies depending on the number of fall events in a circle as well as the number of fall events that are registered in a group by the server.

The major constrain could be the circumstance where a single client is resident in the building during the earthquake. There is no alghrithm in the current system to identify such kind of situation when the building collapses as we define that all the single events as an isolated event which will be excluded from the server’s consideration.

The port in the university for the server is blocked by the firewalls.It took long time to make server port working fine.

**Implementation process**

The implementation phase is dedicated for prototyping and system architecture design.The various component of our system is developed in parallel and is aggregated together at the end.

Discuss the functionalities we need,and

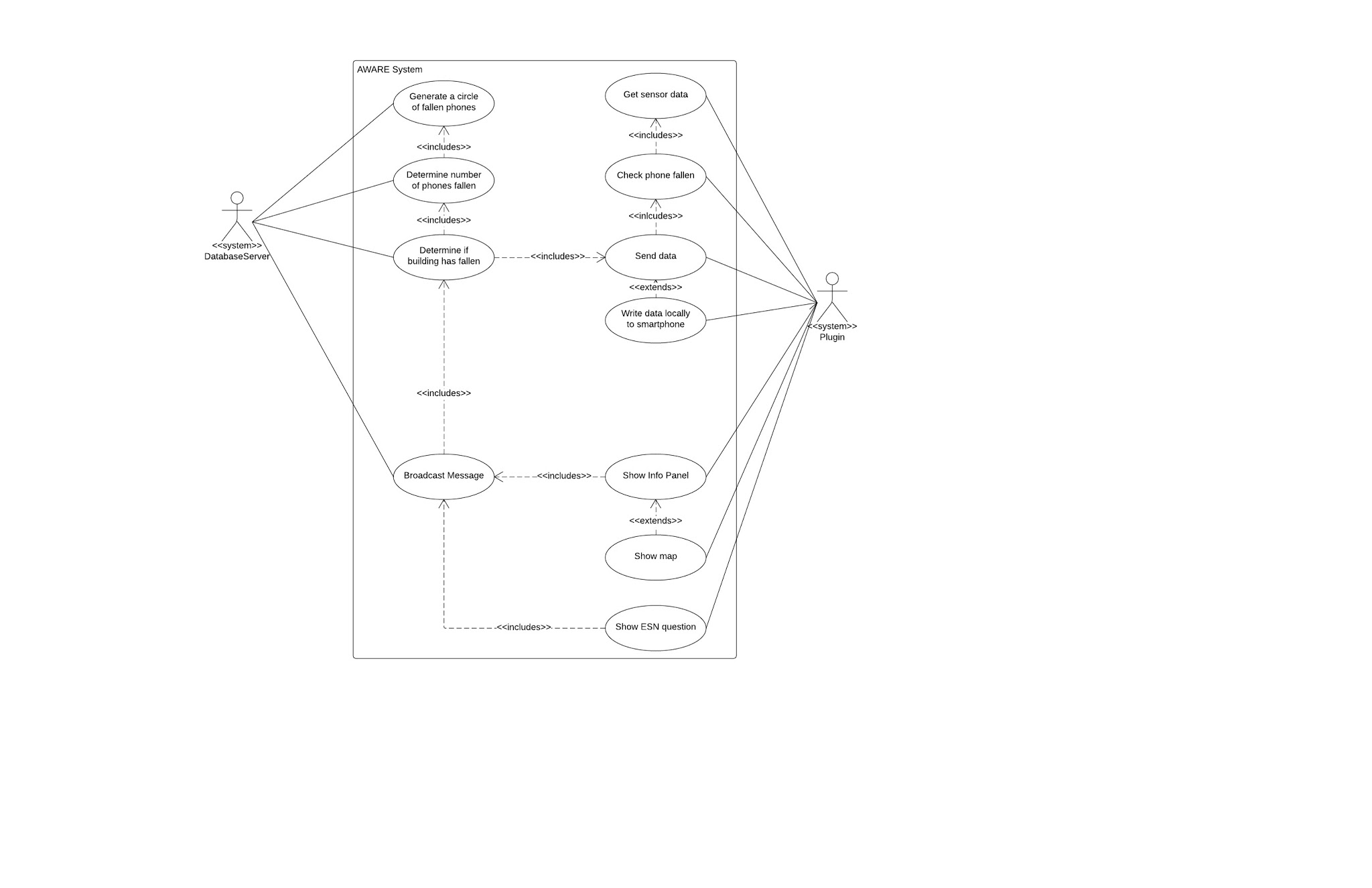
# **Software architecture**

## This chapter describes in detail the implementet software architectures in both phone plugin and in server software. Appropriate UML-diagrams are presented to visualize the architechtual decisions made in design and implementation phases.

|  |  |
| --- | --- |
|  |  |

### Use Cases

When we consider the Plugin and the Server as the actors of our system, we can attach use-cases (or modular functionalities) to each. Figure 7 shows the use case diagram of the system.



|  |  |
| --- | --- |
|  |  |
|  |  |

Using the boiler-plated template, we can determine the SONs as follows:

1. The <database server> shall be able to <generate circles of fallen phones> <when a fallen phone is detected>.

2. The <database server> shall be able to <determine the number of fallen phones> <within its accuracy limits>.

3. The <database server> shall be able to <determine building collapse> <within its accuracy limits>.

4. The <database server> shall be able to <broadcast messages> <when a building collapse is detected>.

a. The <database server> shall be able to <broadcast message> <within ‘t’ minutes of the event>.

5. The <plugin> shall be able to <read the sensor data> <always>.

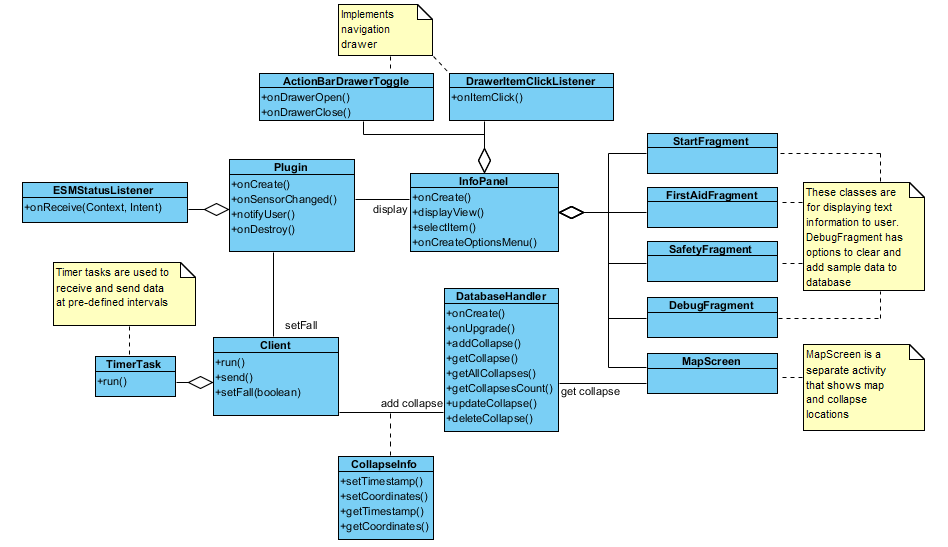
6. The <plugin> shall be able to <check the phone fallen status> <always>

7. The <plugin> shall be able to <show Info Panel> <when the server broadcasts or it is opened from launcher>.

8. The <plugin> shall be able to <show ESN question><when the fall event happens>

### Class diagram

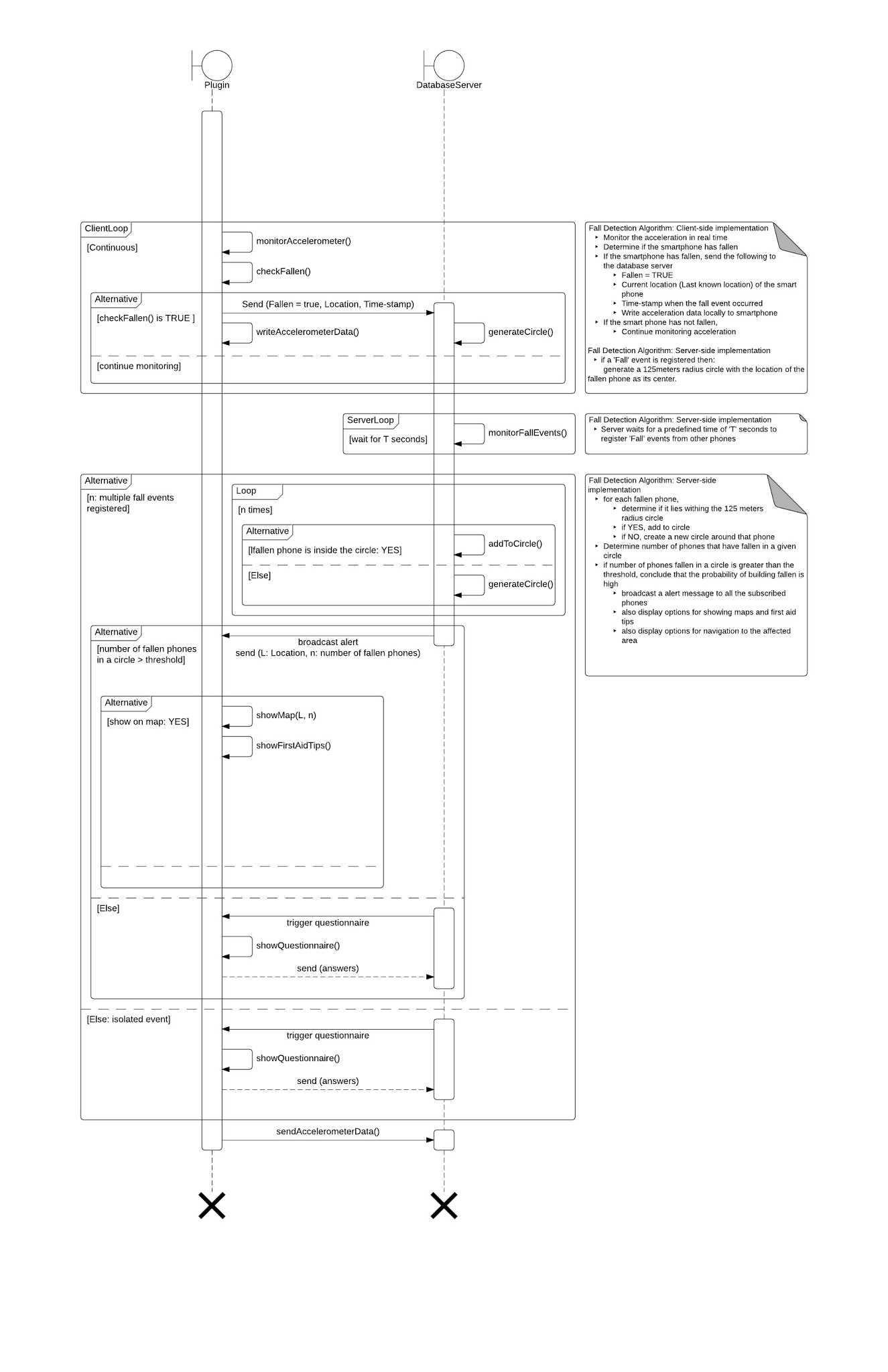
A class diagram represents the static view of an application. We used the class diagram for visualizing, describing, and documenting different aspects of our system. It shall also pave the way for constructing executable code of the software application. Figure 8 shows the class diagram of the system which describes the attributes and operations of the classes of the system.

**Smartphone software’s class diagram**

### 

### Sequence diagram

UML sequence diagrams model the flow of logic within the system in a visual manner. Figure shows the sequence diagram of our system. It enables us to both document and validate your logic. As they represent the dynamic model of the system, it greatly focuses on identifying the behavior within our system.



**Component architecture**

We are building a smartphone based open sensor network that is capable of detecting the collapse of buildings in near real time. It will enable us to, (a) alert smartphone users and Emergency Response Units (ERUs) of any collapsed buildings, (b) provide navigation details to the scene, (c) functionalities to report live events from the scene, and (d) a web based user interface for scientists and researchers to access the accelerometer data of the smartphones subscribed to the system. Ours is a challenging study, as smartphones can be used in myriad ways by their owners. The unpredictable component associated with humans makes it difficult to use constants during system design. Below Figure 5 shows the overview of our system. There are two primary components of our system, Clients – Smartphones, and Server – Database server. This two tier architecture forms the entire Collapse Detection System.

|  |  |
| --- | --- |
|  |  |
|  |  |

Figure 5. Overview of the Collapse Detection System

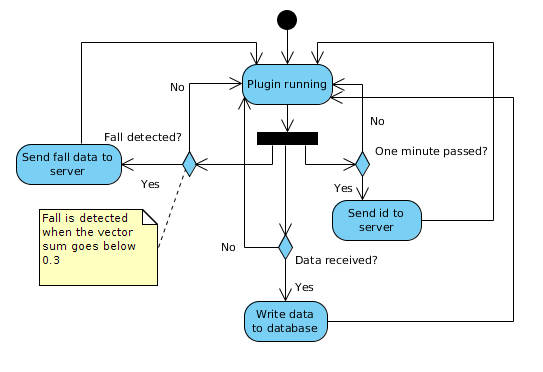
|  |  |
| --- | --- |
| Smartphones The smartphone software will consist of following components:   * **Client** is used to communicate with the server. Client includes the components to send the device id once in a minute, send fall event, and receive data from server. * **Plugin** includes fall detection algorithm and initializes the client when plugin is started. * **Info panel** implements user interface and also displays map where recieved coordinates will be shown.   plugin_components_2.PNG AWARE Framework We shall be using the AWARE platform to design our application. AWARE is a mobile instrumentation middleware aimed at facilitating our understanding of human behavior [24]. AWARE framework is open-sourced and is aimed at building context-aware applications, collect data, and study human behavior. It can mitigate researchers’ effort when building mobile data-logging tools and context-aware applications by encapsulating implementation details of sensor data retrieval. Our project uses AWARE framework to achieve access its Accelerometer and Gyroscope sampling features to obtain raw data from the phone’s sensors. AWARE infrastructure is shown in Figure 6.  Our client software (henceforth referred to as ‘Plugin’) will use the AWARE framework and run on the users smartphones. The server component of our system will run on the AWARE database server that collects all the registered smartphone sensor data. |  |

**System behavior**

**Client**

* The client plugin software has the following functionalities:
* Sensing data on the phones by utilizing its accelerometer, GPS-location, and precise time data.
* Monitoring accelerometer values and detecting a fall.
* Sending timestamp, devide id and location(latitude and longitude) to server when fall event happens.
* Sending the device id once in a minute to the server.
* Providing first aid and safety tips
* Receiving data broadcasted by the server
* Saving the data into a database
* Reading values from database and displaying them on the map when it is opened

**Activity diagram for running plugin**



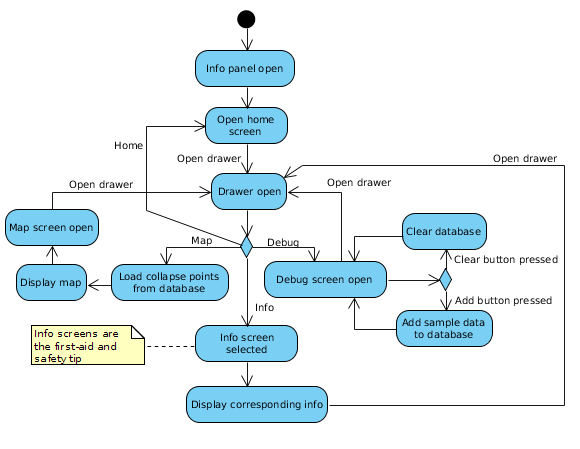
This activity diagram displays the background process of detecting fall events, keeping the server informed that device is still online, and the data receiving function.

When the plugin is activated in AWARE-application it enters the “Plugin running” activity. After that it starts monitoring these three aspects:

* Has the phone fallen.
* Has it been one minute since the last id being sent to server.
* Has the plugin received any data from the server.

The corresponding actions to the aforementioned events are displayed in the diagram. This diagram does not include the ending point hence it is entered only when the plugin is deactivated from AWARE-application.

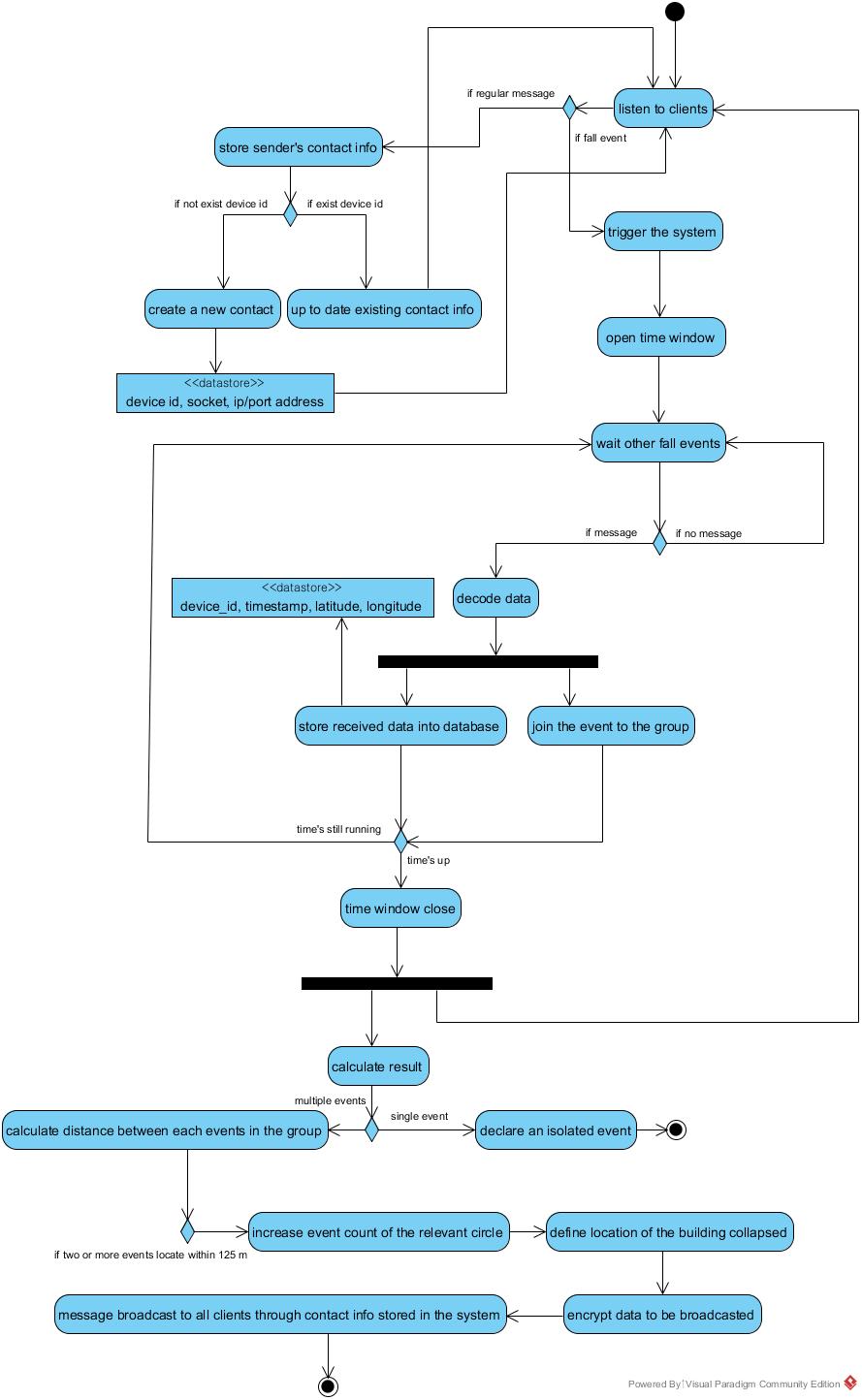
**Sequence diagram for plugin user interface**



This diagram displays the activities that can be executed in the plugins user interface. The diagram is separate from the plugin diagram because either one of these processes can be started separately.

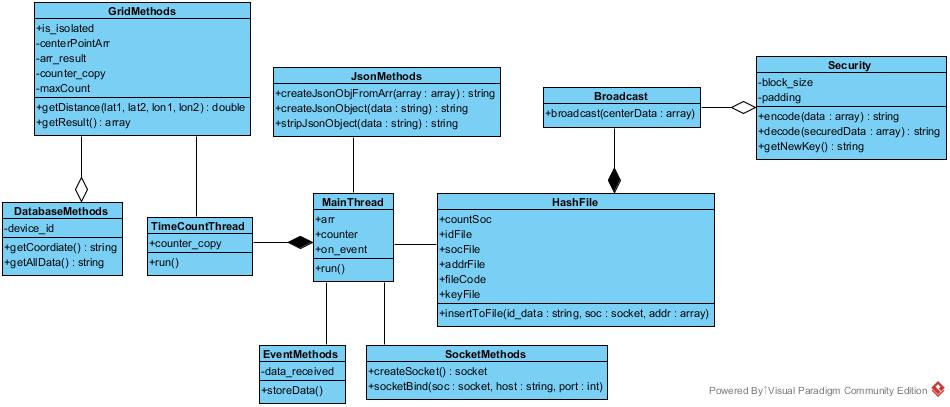
The user interface consist of following functionalities:

* Providing navigation drawer where user can select which information he/she wants to see.
* Displaying the first-aid and safety tip screens (referred to as info screens)
* Displaying the debug screen where the user can clear the received collapse entries from the phones database, and add some sample data in to the databaseto demonstrate how building collapses could appear on map.
* Displaying the map, focusing it over users location fetching collapse locations from database and displaying those locations on the map.

**Server**

**how the system works from server side, along with activity diagram**

1. server starts listening to clinets once the system is on and it waits for any type of data sent from clients
2. if the server receives regular message it stores contact information (socket, ip/port address and shared key) into an array and keep on listening to incoming clients
3. if the server receives a fall event with device id, timestamp and coordinate data it trigger the defining building collapse algorithm
4. the algorithm starts with opening timewindow of 60 seconds
5. server collects all the fall events coming within the timewindow and create a group of clients
6. once the fall event with data has arrived the server decodes it by using shared key and then store it into the database as well as creating a group of fall events
7. if the timewindow is still open then the server returns itself back on listening to other fall events otherwise it begins the building collapse algorithm or announce that the server has detected an isolated event if it was a single event
8. the algorithm is straightforward that the server calculates distance for each of the fall events in a group by using coordinate and increase the count when two events locate within 125 physical meters
9. while the data received has been being analysed the server returns back to listening clients to trigger another event
10. once the algorithm is done calculation server assume that the location of the building has collased was near the client’s coordinate with highest count value in the group and it comes with certain persentage of probability that’s depending on the entire number of events in the group
11. when the calculation is done server starts to prepare sending broadcast messages to all clients those the server has contact information
12. all data before to be broadcasted will go through encryption using each client’s secret key
13. server finally terminates the calculation by broadcasting the defined coordinate of building collpased to all clients

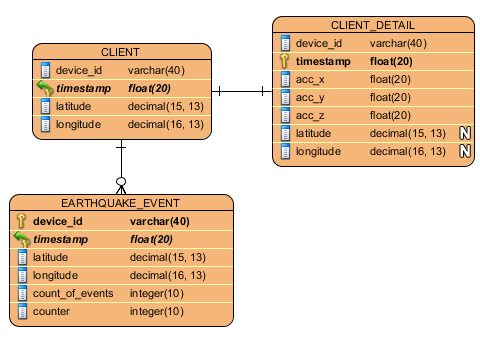


* Realize the UDP communication between the application and server
* Develop a small prototype of UDP communication in both client and server side
* Server setting, installing tools and libraries
* Server can receive data including timestamp,devide id and location(latitude and longitude) which is sent from the Plugin
* Create database and tables in the sever
* Implement database methods
* Implement json methods
* Store received data into local database in the server
* Create and break timewindow using multi-threading
* Implement calculation algorithm of getting distance between two coordinate points
* Implement calculation algorithm of defining coordinate where the building collapsed
* Store and maintain client’s contact information using file directories
* Implement broadcating messages
* Implement data encryption and decryption
* Conduct module test

**Data structures**

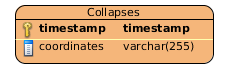
**Server**

The system contains two databases.One is the server database.One is the client database.For the server database, the data is sent in JSON format.And they are stored in mysql with ‘ timestamps’ as primary key .It is stored in persistent storage in mysql.The database schema is the physical database schema.This schema pertains to the actual storage of data,and its form of storage like files,indices etc.It defines that how data will be stored in secondary storage etc.

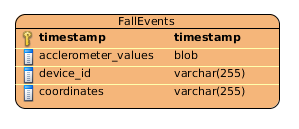


**Client**

In client the received collapse coordinates and the time of the collapse are inserted in a internal SQLite database file. Addition happens by the time the collapse-message arrives. Thetimestamp is used as the primary key in the database. This table is displayed in following diagram.



It could also be possible to implement another database where to store the recorded data related to a fall event detected by the phone. This data can be then later sended to the server for further review when testing the system’s functionality. This table would also have the timestamp as a primary key and it would include the recorded accelerometer values before and after the fall event, device id, and the time of the detected fall.



**Hardware documentation**

**Phones**

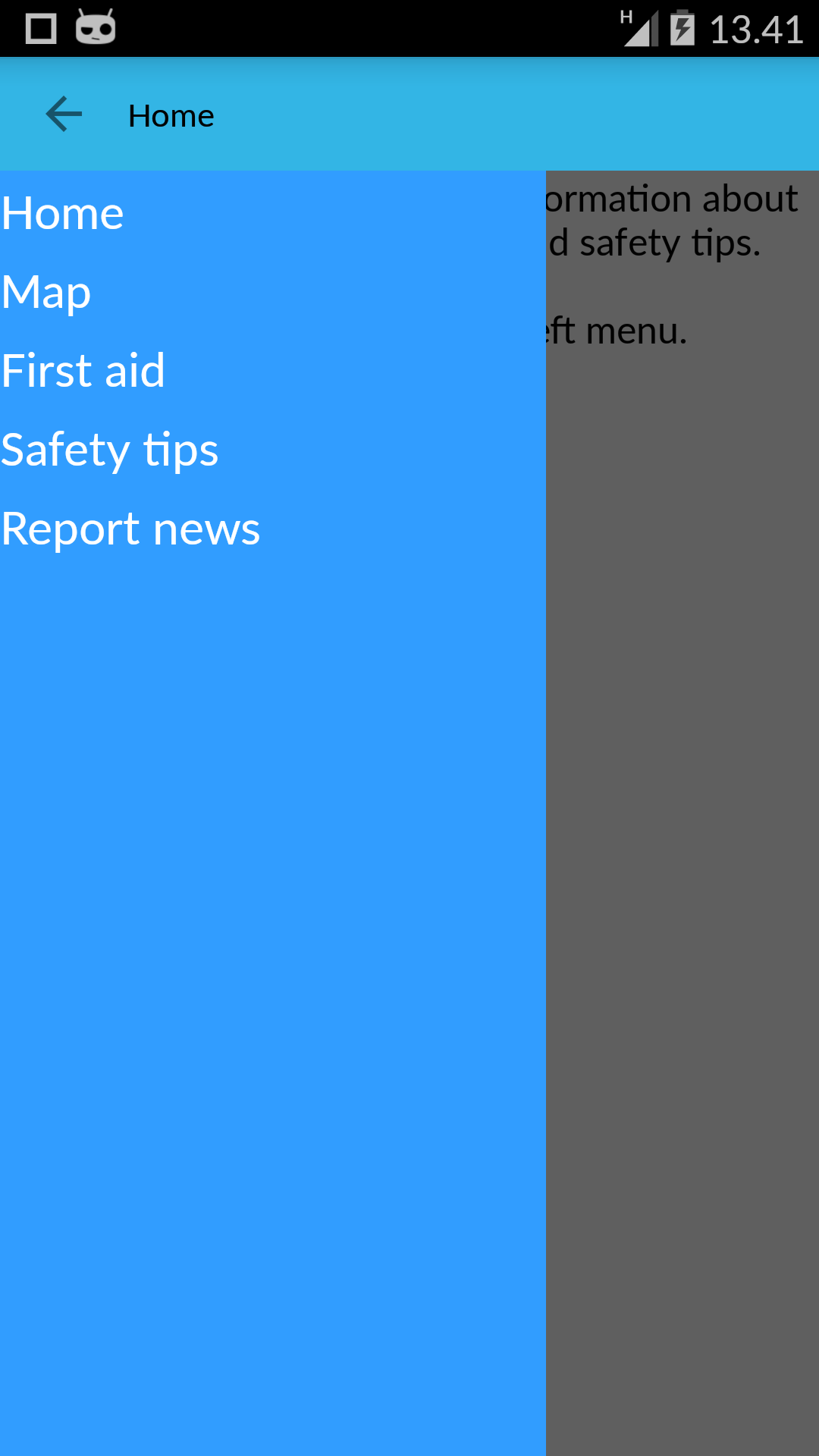
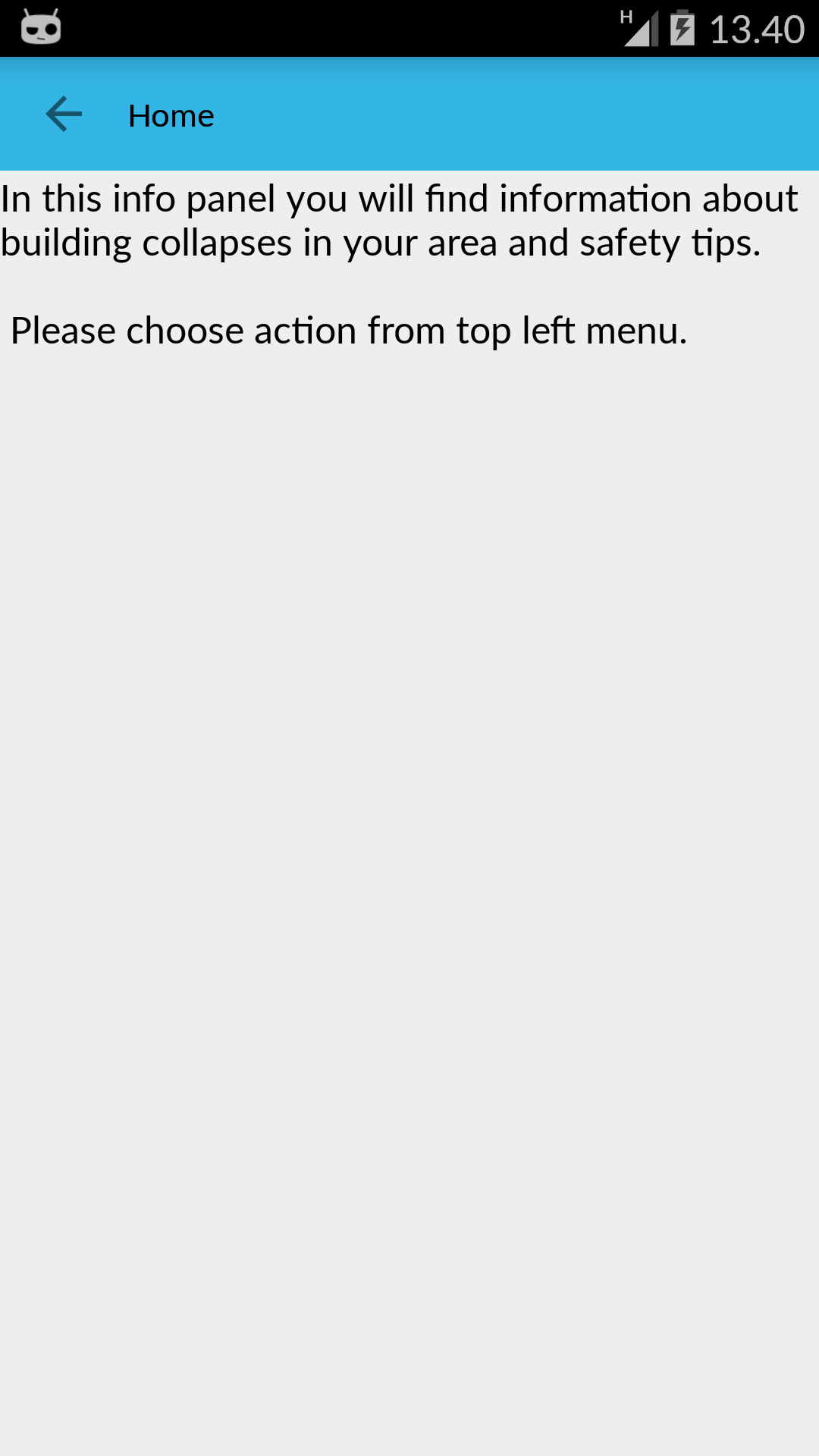
The systems client software will be deployed for Android phones. These phones are required have the following services to be able to monitor and detect fall events:

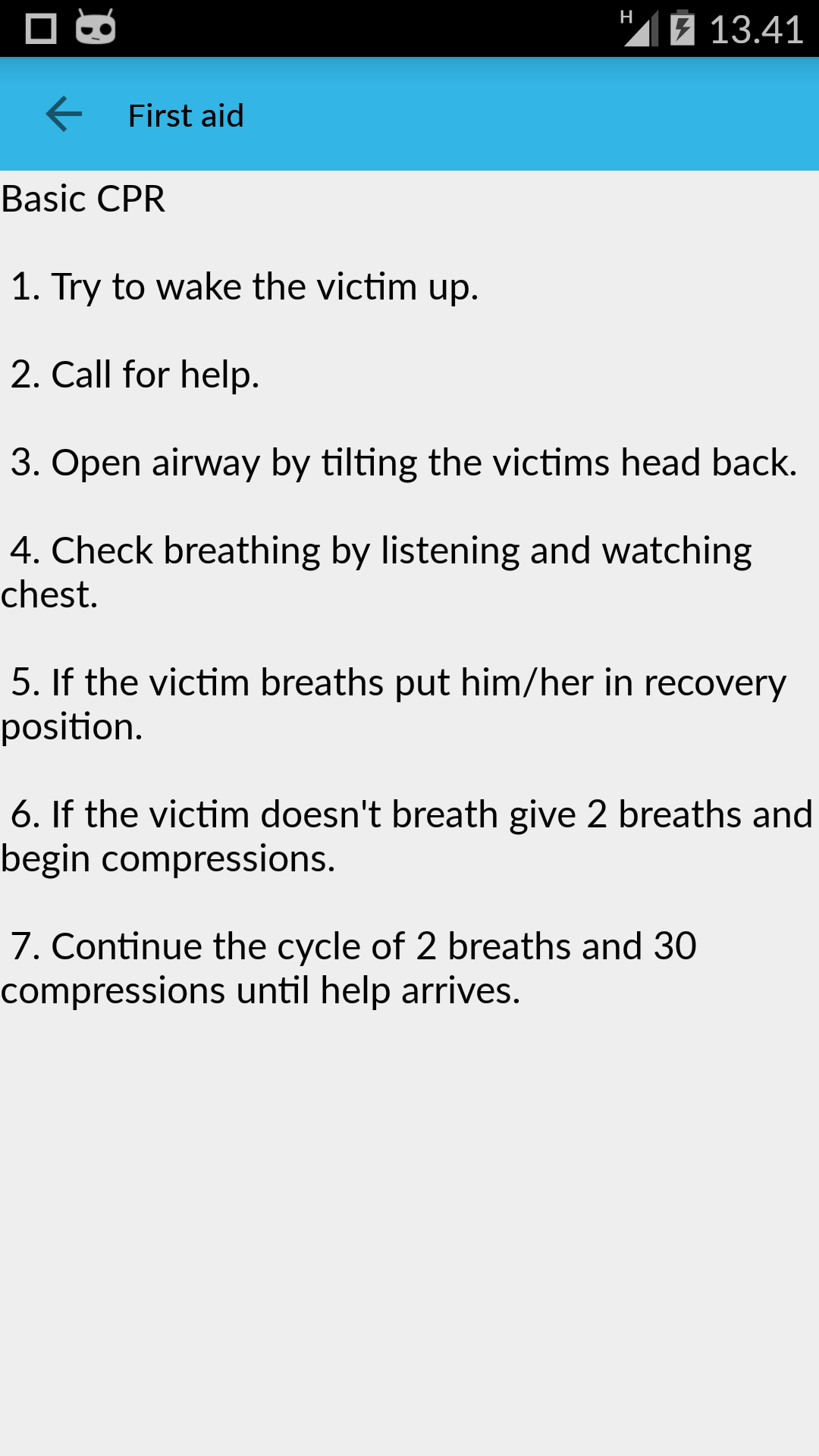
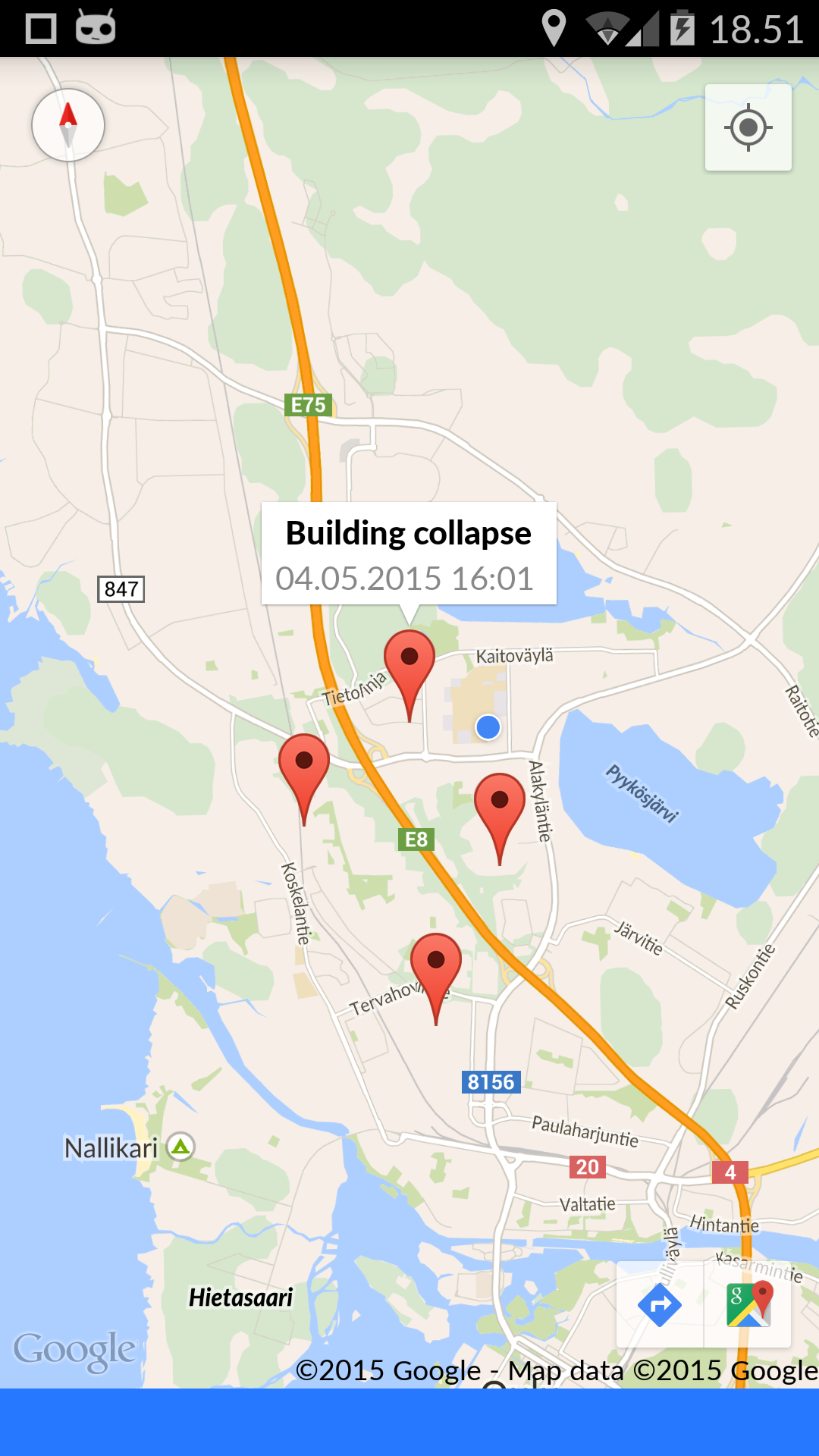
* Accelerometer
* GPS
* Wifi-connection

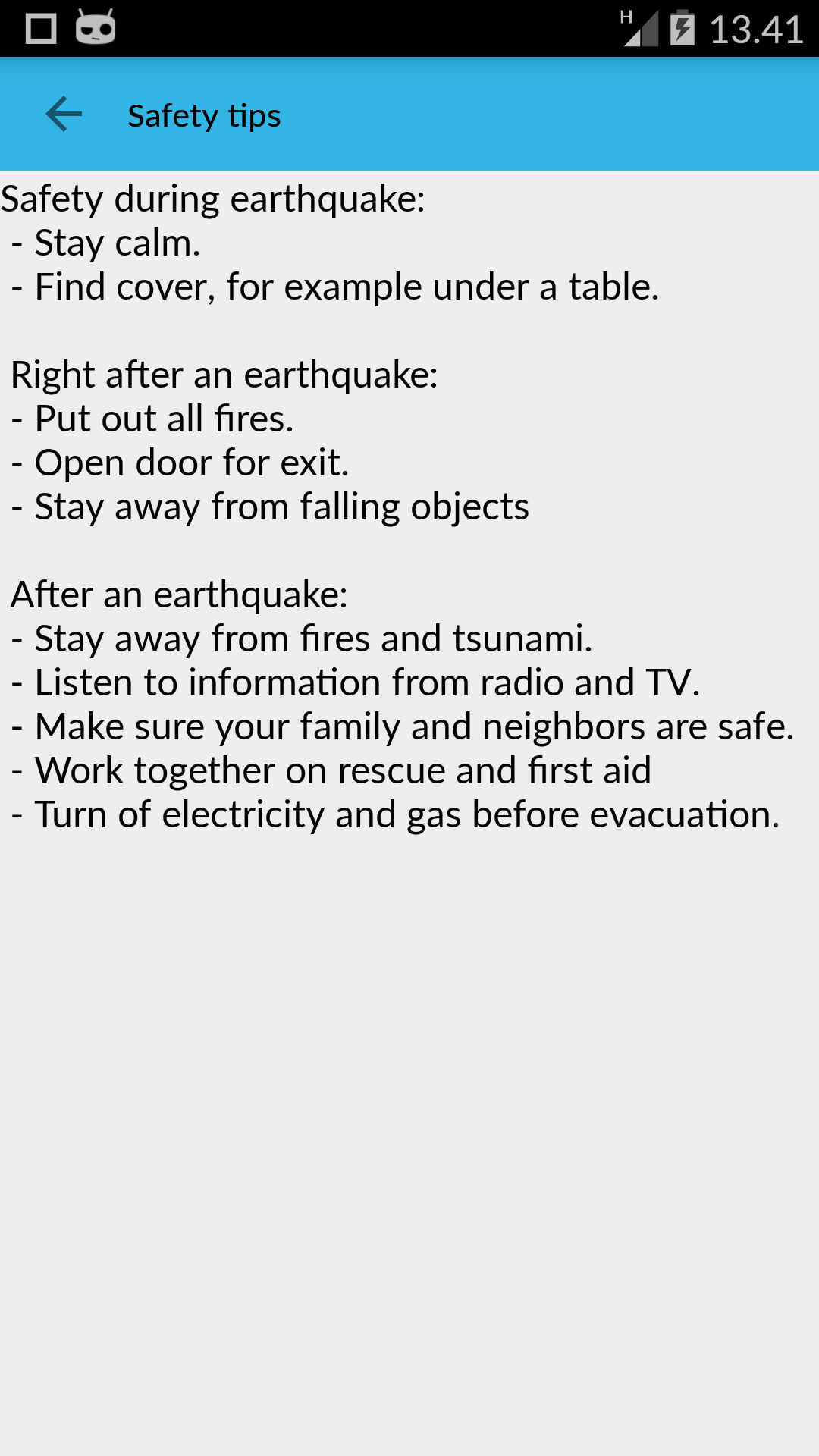
**User interface**

**Mobile UI**

The mobile user interface is designed to have all necessary functionality available easily for the user. The UI includes home screen, map, first aid tips, safety tips and debug screen. The proposed news reporting functionality is not implemented at this point.







**Third party materials**

**Server side**

The server ‘s account is [vm0101.virtues.fi](http://vm0101.virtues.fi/)

Linux version is 2.6.32-504.8.1.e16.x86\_64

gcc version is 4.4.7 20120313(Red Hat 4.4.7-11)

Python version is 2.6.6

server version:apache/2.2.15(Unix),the server which is built in October 16 2014 14:48:21

**Libraries**

Mysql version is 5.1.73

pycrypto version is 2.6.1

socket

time

numpy

math

base64

**Client side**

Android studio

aware version is 3.3.3

android version is 2.3.3 or higher

google map API

facebook API (will be implemented later)

**Security and privacy**

In this project we would like to ensure the user’s privacy and security. When the data is traveling between client and server it requires data encryption.

The usage of an encryption algorithm(AES).

We will use a shared key to encrypt and to decrypt the data.

**Project risk assessment**

One of the member’s leaving has made the workload heavy.

The team lacks experience in some areas of the project.

**To do**

time synchronization

post news using the facebook API

Write the data locally for plugin

**Reference**

**Contribution**

All members equally participated in the certain team meetings and helped each other during the within-team deliverables. There are other meetings for client side and server side separately.Below is a breakdown of the contribution of each team member.

## Team Meetings:

3 weeks: 2 hours each,2 times meeting per week - 12 hrs. All team members were present during the meeting.

5 weeks:2 hours each -10 hrs.Perttu and Zeyun were present during the meeting for the client side.

5 weeks:2 hours each -10 hrs.Haejong and Zeyun were present during the meeting for the server side.

## Individual Contributions

### Zeyun Zhu

I am the project manager of the team and have been entrusted with some additional project management duties. It was my duty to ensure that we have proper team meetings every week. I also charted our plan for the implementation stage, having provided ideas on how to implemente the project ideation. As a team member, I contributed through setting up server, UDP communication between client and server, sending the data from client to the server, deformat the data which is sent by the client in the server,doing the database and store the data into database,making broadcast message brainstorming , doing XMPP protocol and testing including the client,database,server. Below is a breakdown of my total effort during this stage.

|  |  |
| --- | --- |
| Topic | Hours |
| Project Management | 6 |
| Meeting | 32 |
| Server setting | 6 |
| UDP communication | 12 |
| Client implementation | 10 |
| Database implementation | 8 |
| Test | 28 |
| XMPP protocol | 12 |
| Report writing |  |
| **Total** |  |

Table 4. Zeyun Zhu working hours

### Haejong Dong

I have contributed on server system implementation mainly. I took part in implementation of clients and server system communication and database design, implementation. I am responsible for the implementation of server side of the system. For more details about my works are blow in the table

|  |  |
| --- | --- |
| Topic | Hours |
| Meeting | 22 |
| Server side implementation | 30 |
| Client, Server communication | 10 |
| Report | 5 |
| Database | 10 |
| System testing | 5 |
| **Total** | **82** |

Table 5. Hejong Dong working hours

### Perttu Pitkänen

I contributed to the project by implementating the client. I am responsible for implementing the fall event detection algorithm, the info panel user interface and data communnications in the plugin. Below is a breakdown of my total effort during this stage.

|  |  |
| --- | --- |
| Topic | Hours |
| Meeting | 22 |
| Fall event detection algorithm | 10 |
| UI design and implementation | 23 |
| Udp communication | 20 |
| Writing report | 13 |
| Phone database | 6 |
|  |  |
| **Total** |  |

Table 6. Perttu Pitkänen working hours

## Summary

The table summarizes the contribution of each of the team members to this implementation phase.

|  |  |  |
| --- | --- | --- |
| Name | Total Work in Hours | Contribution to the Total (%) |
| Zeyun |  |  |
| Haejong |  |  |
| Perttu |  |  |
| **Total** |  |  |

Table 8. Summary table of total work load and individual contributions