NextWave Helper: Android App for Search and Rescue

CS492 Project

MDL15CS016	CSU152010	Anand Shekhar
MDL15CS048	CSU152048	Joe Davis
MDL15CS103	CSU152049	Soumya M.
CEC15CS047	CSU172067	Saran Narayan

B. Tech Computer Science & Engineering

Department of Computer Engineering
Model Engineering College
Thrikkakara, Kochi 682021
Phone: +91.484.2575370
http://www.mec.ac.in
hodcs@mec.ac.in

June 2019

Model Engineering College Thrikkakara
Department of Computer Engineering

C E R T I F I C A T E

This is to certify that, this report titled ***NextWave Helper*** is a bonafide record of the work done by

MDL15CS016 CSU152010 Anand Shekhar
MDL15CS048 CSU152048 Joe Davis
MDL15CS103 CSU152049 Soumya M.
CEC15CS047 CSU172067 Saran Narayan

Eighth Semester B. Tech. Computer Science & Engineering

students, for the course work in **CS492 Project**, which is the second part of the two semester project work, under our guidance and supervision, in partial fulfillment of the requirements for the award of the degree, B. Tech. Computer Science & Engineering of **Kerala Technological University**.

Guide

Manilal D. L
Head of Department
Computer Engineering

Coordinator

Manilal D L
Associate Professor
Computer Engineering

Head of the Department

Manilal D L
Associate Professor
Computer Engineering

June 7, 2019

Acknowledgements

This project would not have been possible without the kind support and help of many individuals. We would like to extend my sincere thanks to all of them.

First of all, We would like to thank our esteemed Principal, Prof. (Dr.) Vinu Thomas, for his guidance and support in maintaining a calm and refreshing environment to work in and also for providing the facilities that this work demanded.

We are highly indebted to our Project Coordinator and Head of the Department, Manilal D L, Associate Professor for their guidance, support and constant supervision throughout the duration of the work as well as for providing all the necessary information and facilities that this work demanded.

We would like to thank our Project Guide, Manilal D L for his support and valuable insights and also for helping me out in correcting any mistakes that were made during the course of the work.

We offer our sincere gratitude to all our friends and peers for their support and encouragement that helped me get through the tough phases during the course of this work.

Last but not the least, we thank the Almighty God for guiding me through and enabling us to complete the work within the specified time.

Anand Shekhar
Joe Davis
Soumya M.
Saran Narayan

Abstract

Natural and man-made disasters can have devastating effects on human society, but they can largely be mitigated with the coordinated efforts of rescue workers. However, existing disaster response means have multiple obstacles like accessibility, ease of use, dependency on social media and requirement of special skill sets on the part of the public. There can also be cases when the whole network infrastructure fails, which can seriously hinder rescue efforts and endanger the lives of the affected.

To overcome these limitations we developed an Android-based application that offers user-friendliness and real-time data related to crowd-sourced information, which can provide missing persons data, show affected areas and relief centre locations, and handle requests. In cases when the networks fails we use Mobile Ad-hoc Networks (MANET) that will serve to establish and provide communication and coordination among mobile devices during the emergency situation.

Contents

List of Figures **v**

1 Introduction **1**
1.1 Proposed Project . . . 1
1.1.1 Problem Statement . . . 1
1.1.2 Proposed Solution . . . 2

2 System Study Report **3**
2.1 Literature Survey . . . 4
2.2 Proposed System . . . 7

3 Software Requirement Specification **8**
3.1 Introduction . . . 8
3.1.1 Purpose . . . 8
3.1.2 Intended Audience and Reading Suggestions . . . 8
3.1.3 Project Scope . . . 8
3.1.4 Overview of Developer's Responsibilities . . . 8
3.2 Overall Description . . . 9
3.2.1 Product Perspective . . . 9
3.2.2 Product Functions . . . 9
3.2.3 User Classes and Characteristics . . . 9
3.2.4 Operating Environment . . . 10
3.2.5 Design and Implementation Constraints . . . 10
3.2.6 General Constraints . . . 11
3.2.7 Assumptions and Dependencies . . . 11
3.3 External Interface Requirements . . . 11
3.3.1 User Interfaces . . . 11
3.3.2 Hardware Interfaces . . . 11
3.3.3 Software Interfaces . . . 11
3.3.4 Communication Interfaces . . . 12
3.4 Hardware and Software Requirements . . . 12
3.4.1 Hardware Requirements . . . 12
3.4.2 Software Requirements . . . 12
3.5 Functional Requirements . . . 13
3.5.1 View announcements . . . 13
3.5.2 View missing persons database . . . 13

3.5.3 Mark affected areas on the map . . . 13
3.5.4 Unmark affected areas on the map . . . 13
3.5.5 Request for help along with geolocation . . . 14
3.5.6 View the help requests and locations . . . 14
3.5.7 Text communication among the victims and rescue workers . . . 14
3.5.8 View Relief centre locations . . . 14
3.5.9 Login for rescue coordinator . . . 14
3.5.10 Update the missing persons database . . . 15
3.5.11 Update announcements . . . 15
3.5.12 Update relief center locations and donation sites . . . 15
3.5.13 Update help requests status . . . 15
3.6 Non-functional Requirements . . . 16
3.6.1 Performance Requirements . . . 16
3.6.2 Security Requirements . . . 16
3.6.3 Software Quality Attributes . . . 16
3.7 Other Requirements . . . 17

4 System Design **18**
4.1 System Architecture . . . 18
4.1.1 GPS module . . . 19
4.1.2 Database server . . . 19
4.1.3 Web portal . . . 19
4.2 Input Design . . . 19
4.2.1 Use Case Diagram . . . 19
4.2.2 View announcements . . . 21
4.2.3 View missing persons database . . . 22
4.2.4 Mark affected areas on the map . . . 23
4.2.5 Unmark affected areas on the map . . . 24
4.2.6 Request for help along with geolocation . . . 25
4.2.7 View the help requests and locations . . . 26
4.2.8 Text communication among the victims and rescue workers . . . 27
4.2.9 View Relief centre locations . . . 28
4.2.10 Login for rescue coordinator . . . 29
4.2.11 Update the missing persons database . . . 30
4.2.12 Update announcements . . . 31
4.2.13 Update relief center locations and donation sites . . . 32
4.2.14 Update help requests status . . . 33
4.3 Database Design . . . 33
4.4 Class Diagram . . . 35
4.5 Activity diagram . . . 36
4.5.1 View announcements . . . 36
4.5.2 View missing persons database . . . 37
4.5.3 Mark affected areas on the map . . . 38
4.5.4 Unmark affected areas on the map . . . 39
4.5.5 Request for help along with geolocation . . . 40
4.5.6 View the help requests and locations . . . 41

4.5.7 Text communication among the victims and rescue workers . . . 42
4.5.8 View Relief centre locations . . . 43
4.5.9 Login for rescue coordinator . . . 44
4.5.10 Update the missing persons database . . . 45
4.5.11 Update announcements . . . 46
4.5.12 Update relief center locations and donation sites . . . 47
4.5.13 Update help requests status . . . 48
4.6 Libraries and Packages Used . . . 49
4.6.1 Retrofit . . . 49
4.6.2 Volley . . . 49
4.6.3 Mapbox Maps Android SDK . . . 49
4.6.4 Dagger . . . 49
4.6.5 Gson . . . 49
4.6.6 Room Persistence Library . . . 50
4.6.7 Nearby Connection API . . . 50
4.6.8 Lumen . . . 50
4.7 Module Description . . . 51
4.7.1 Web Interface . . . 51
4.7.2 API . . . 51
4.7.3 Android Application - Normal mode . . . 51
4.7.4 Android Application - Disaster mode . . . 51

5 Data Flow Diagram 52
5.1 Level 0 DFD . . . 52
5.2 Level 1 DFD . . . 53
5.3 Level 2 DFD . . . 54

6 Implementation 56
6.1 Algorithms . . . 56
6.1.1 Setup P2P Connection . . . 56
6.1.2 P2P MANET reactive routing . . . 57
6.2 Development Tools . . . 57
6.2.1 Git . . . 57
6.2.2 XAMPP/LAMP . . . 58
6.2.3 Visual Studio Code . . . 58
6.2.4 Android Studio . . . 58
6.2.5 Postman . . . 58

7 Testing 59
7.1 Testing Methodologies . . . 59
7.2 Unit Testing . . . 59
7.3 Integration Testing . . . 59
7.4 System Testing . . . 61

8 Graphical User Interface 62
8.1 GUI Overview . . . 62

8.2 Main GUI Components 62
8.2.1 Website 62
8.2.2 Android Application 65

9 Results **68**

10 Conclusion **71**

11 Future Scope **72**

12 Publication **73**

References **82**

List of Figures

Figure 3.1: Use Case Diagram 1 . . . 13
Figure 3.2: Use Case Diagram 2 . . . 14
Figure 3.3: Use Case Diagram 3 . . . 15
Figure 3.4: Use Case Diagram 4 . . . 15

Figure 4.1: System Architecture . . . 18
Figure 4.2: Use Case Diagram . . . 20
Figure 4.3: Database Schema . . . 34
Figure 4.4: Class Diagram . . . 35
Figure 4.5: View announcements . . . 36
Figure 4.6: View missing person database . . . 37
Figure 4.7: Mark affected areas on the map . . . 38
Figure 4.8: Unmark affected areas on the map . . . 39
Figure 4.9: Request for help along with geolocation . . . 40
Figure 4.10: View the help requests and locations . . . 41
Figure 4.11: Text communication among the victims and rescue workers . . . 42
Figure 4.12: View Relief centre locations . . . 43
Figure 4.13: Login for rescue coordinator . . . 44
Figure 4.14: Update the missing persons database . . . 45
Figure 4.15: Update announcements . . . 46
Figure 4.16: Update relief center locations and donation sites . . . 47
Figure 4.17: Update help requests status . . . 48

Figure 5.1: DFD Level 0 . . . 52
Figure 5.2: DFD Level 1 . . . 53
Figure 5.3: DFD Level 2 : Database updations and Normal Mode Process . . . 54
Figure 5.4: DFD Level 2 : Disaster Mode Process . . . 55

Figure 7.1: Trying to call the API without the authorization header . . . 60
Figure 7.2: The Android Application getting data from the API . . . 60

Figure 8.1: Login page . . . 62
Figure 8.2: Home page . . . 63
Figure 8.3: Announcements . . . 63
Figure 8.4: Relief Centre . . . 64
Figure 8.5: Help requests . . . 64
Figure 8.6: Intro slides . . . 65

Figure 8.7: Welcome screen 65
Figure 8.8: Dashboard 66
Figure 8.9: Missing person reports 66
Figure 8.10: Group chat 67
Figure 8.11: Location sharing 67

Figure 9.1: Help request section 68
Figure 9.2: Request for help 68
Figure 9.3: New help request 69
Figure 9.4: Rescue worker 69
Figure 9.5: Help request assigned by rescue coordinator 70
Figure 9.6: Help request assigned 70

Chapter 1

Introduction

In mid August 2018, Kerala, a state in India was hit by one of the worst floods in its recent memory. With over 483 dead and more than a million displaced, it remains a dark specter that looms over the state even today. 3,200 relief camps were set up during and after the floods. The property damage is estimated to be around Rs.40,000 crore (or US $5.6 billion). Soon after the tragedy struck, technical experts and volunteers came together and built the website 'keralarescue.in'[2][3] to obtain a platform to connect online calls for help with people coordinating offline rescue operations. The website played an integral role in search and rescue operations and helped saved lives.[4]

The website had a plethora of features, which we took inspiration from, like showing the latest announcements, help requests registered by people affected by the flood, locations of relief camps, and a list of missing and found people. However, being built in a hurry, the website was not as well designed as it could have been. The user interface was cluttered and not mobile ready, this affected the usability and made it difficult for senior citizens, who were primarily using their mobile phones to navigate the website.

1.1 Proposed Project

Currently, communication in disaster-affected areas can be erratic, with damage or destruction of the communication infrastructure. At the same time, it is possible that communication networks are fine. In order to remedy this, we will create an android application which has 2 modes of operation - a normal mode and a disaster mode. The normal mode will allow communication through normal channels, while the disaster mode will rely on an ad-hoc peer to peer network in order to allow communication even when infrastructure is damaged. This will allow the end-user to choose how to operate the application, based on the current situation.

1.1.1 Problem Statement

Natural and man-made disasters can have devastating effects on human society, but they can largely be mitigated with the coordinated efforts of rescue workers. However, existing disaster response means have multiple obstacles like accessibility, ease of use, dependency on social media and requirement of special skill sets on the part of the public. There can also be cases when the whole network infrastructure fails, which can seriously hinder rescue efforts and endanger the lives of the affected.

1.1.2 Proposed Solution

This project aims at creating a mobile application, which can be used to prepare for, as well as mitigate the aftereffects of a disaster. During a disaster, one of the most important things is information. Incorrect or insufficient information during a disaster can have fatal consequences for the affected victims. To overcome these limitations we plan to develop an Android-based application, which are now used by the vast majority of people[5][6] that offers user-friendliness and real-time data related to crowdsourced information(shown to be a useful method to obtain information during disasters[7][8]), which can provide missing persons data, show affected areas and relief centre locations, and handle requests. This project will also improve the situation in cases where the networks fails, by having the capability to create a MANET[9][10][11] (Mobile Ad-Hoc Network) for communication in cases where the communication infrastructure is down, while also being able to use the normal infrastructure for purposes such as crowdsourcing.

Chapter 2

System Study Report

Other disaster search and rescue applications have already been created. A detailed comparison showing the advantages and disadvantages of the disaster search and rescue projects considered for this project is shown in the table, which we used to refine our application.

2.1 Literature Survey

Title	Author	Year	Advantages	Disadvantages
Android App using MANET for Search and Rescue Operation During Disaster	Vitri T, et al.	2017	-Uses peer-to-peer network for group communication. -Locating peers on the map.	-Requires users to enter network address to login before connecting to the other peers
Mobile Devices Routing Using Wi-Fi Direct Technology	Ricardo Pagoto Marinho, et al.	2015	-Flooding is the simplest. -AODV is reactive, hence efficient. -LAR reduces control overhead and bandwidth usage.	-Flooding consumes more energy. -AODV does not check route periodically, the delay is more. -Availability of GPS may not be guaranteed.
Comparison between AODV and DSDV Routing protocols in Mobile Ad-hoc Network	Afrah Daas, et al.	2015	-DSDV is suitable for larger networks, whereas AODV is suitable for smaller networks. -DSDV keeps tracks of the nodes, hence the delay is less.	-DSDV consumes more bandwidth due to continuous broadcast.
Geographic and Reactive Routing Protocols for MANET	Ashutosh Srivastava, et al.	2013	Found that geographic routing is better -Reduced overhead for these protocols	-Might be slower, because they are reactive protocols

Title	Author	Year	Advantages	Disadvantages
Crowdsourced Mobile App for Flood Risk Management	Mary Jane C. Samonte., et al.	2017	-A mobile app developed to disseminate flood information. -Send SMS to the servers to interpret and filter data	-A good network is required. -Server should be capable of handling a lot of traffic.
A Smartphone assisted Post-Disaster Victim Localization Method	Akbar Hossain, et al	2016	-Location coordinates of the smartphones. -Monitors radio environment, disaster detection and victim localization.	-Localization becomes challenging in the case of a non-functional eNB. -Improper network resource allocation.
Twitter in disaster mode: smart probing for opportunistic peers	Theus Hossmann, et al. , et al.	2012	-Two operation modes: normal and disaster. -In the normal operation mode, uses cellular infrastructure -Disaster mode works in a peer-to-peer manner	-This approach uses Bluetooth.

Title	Author	Year	Advantages	Disadvantages
Study on crowdsourcing-compatible disaster information management system based on GIS	Lei Ma, et al.	2014	Based on disaster information collection, crowdsourcing model and VGI (Volunteered Geographic Information)	-Irregularities, such as lack of essential elements and vague description.
Location Based Early Disaster Warning and Evacuation System on Mobile Phones using OpenStreetMap	Rahman, Alam, Chowdhury	2012	-Showed warning of upcoming disasters -Blind people can use it	-Lack of details on OSM -Accurate GPS is required
Development of Smartphone Application for Offline Use in Case of Disaster	Yuze, Qian, Suzuki	2013	-Showed evacuation routes -Showed places with possible flooding	-Disaster manual can be easier to understand -User experience
Disaster Messenger: An Android based Infrastructure Less Application for Post Disaster Information Exchange	Bhattacharjee, Kanta, Modi, Paul, DasBit	2016	Not an energy hog -Can receive data from multiple sources	-User experience -Data mules

2.2 Proposed System

Our proposed system will act as a unified interface for multiple features, and can be used in any sort of disaster to aid the relief efforts. The application will be able to operate in 2 distinct modes:

- Normal mode
- Disaster mode

In normal mode, the application will operate as a normal application, relying on cellular infrastructure, like mobile data and wifi to send and receive messages. The application will have the following features, all of which are crowdsourced:

1. Donation link
2. Missing person locator
3. Relief center locations
4. Affected areas
5. Announcements

In disaster mode, the application will not operate using the normal cellular infrastructure, but using a peer to peer infrastructure. This peer to peer infrastructure will be an ad-hoc network created by mobile devices in the area(a MANET). This will allow mobile devices in the affected area to communicate, even in the event that the communication infrastructure is damaged or inoperational. The features available in disaster mode are as follows:

1. Basic text communication
2. Emergency help request
3. Geo-location sharing

Chapter 3

Software Requirement Specification

3.1 Introduction

3.1.1 Purpose

The purpose of this software requirements specification is to maintain all the functions and specifications of NextWave Helper. It also contains the detailed descriptions for the requirements specified.

3.1.2 Intended Audience and Reading Suggestions

The intended audience of the document are all the stakeholders, testers, users and developers of this project.

3.1.3 Project Scope

This project aims at creating a mobile application, which can be used to prepare for, as well as mitigate the aftereffects of a disaster. During a disaster, one of the most important things is information. Incorrect or insufficient information during a disaster can have fatal consequences for the affected victims. This issue is further exacerbated by the fact that normal methods of communication may be inoperational, due to damage caused by the disaster. This project will improve the situation, by having the capability to create a MANET (Mobile Ad-Hoc Network) for communication in cases where the communication infrastructure is down, while also being able to use the normal infrastructure for purposes such as crowdsourcing.

3.1.4 Overview of Developer's Responsibilities

The developer will be responsible for implementing the following features :

- Portal for displaying the announcements.
- Database for relief camp locations, donation collection centers and missing persons.
- An interface to connect to these databases via HTTP.
- A group chat for the users to communicate.

- Displaying map with the affected areas with information collected from the users using crowd-sourcing.
- An option for the users to share the location and request for help via the app.

3.2 Overall Description

3.2.1 Product Perspective

This application is specifically designed for Android. There needs to be a GPS based system for the application to access. The interface will be made to have a similar look and feel that is consistent with other Android applications. Most Android applications have a similar way to display and navigate through data. This familiar GUI will make the user feel more comfortable navigating and viewing the data on our system.

3.2.2 Product Functions

The project aims to develop an interactive mobile application that can perform the following functions:

- Missing person locator.
- Relief center locations.
- Affected areas.
- Announcements.
- Basic text communication.
- Emergency help request.
- Geo-location sharing.

3.2.3 User Classes and Characteristics

- Disaster victim
- Rescue worker
- Rescue coordinator

Users	Characteristics
Disaster victim	• View announcements. • View missing persons database. • Mark/unmark affected areas on the map. • View relief center locations. • Text communication among the victims and rescue workers. • Request for help along with geolocation.
Rescue Worker	• View announcements. • View the help requests and locations. • Mark/unmark affected areas on the map. • Text communication among the victims and rescue workers.
Rescue Coordinator	• Login to identify as a rescue coordinator • Update the missing persons database. • Add new announcements. • Update relief center locations and donation sites. • View and update help requests.

3.2.4 Operating Environment

- The application will run on the Android mobile operating system. It will support Android versions 7 and above.

3.2.5 Design and Implementation Constraints

- Hardware Limitations: All devices must have a GPS module and a WiFI module.
- Safety and Security Considerations: Only rescue coordinators must be able to modify data.
- Criticality of the Application: The server applications must be available 365 days.

3.2.6 General Constraints

- Battery usage must be limited as far as possible
- Some data must be stored locally to avoid load on server during critical times (like maps).

3.2.7 Assumptions and Dependencies

All the hardware and software requirements of the team to carry out the development activities are not finalised.

3.3 External Interface Requirements

3.3.1 User Interfaces

- When the application is opened for the first time, a welcome screen will be shown, which asks for the persons Name and blood group and is a one-time registration.
- The home screen will show a banner, and other options that can be selected. The options will be as follows:
  - Missing Persons
  - Map
  - Request for Help
  - Disaster Mode

3.3.2 Hardware Interfaces

Since neither the mobile application nor the web portal have any designated hardware, it does not have any direct hardware interfaces. The physical GPS is managed by the GPS module in the mobile phone and the hardware connection to the database server is managed by the underlying operating system on the mobile phone and the web server.

3.3.3 Software Interfaces

The mobile application communicates with the GPS module in order to get geographical information about where the user is located and the visual representation of it, and with the database in order to get the information about affected areas and relief camps.

The android application will be programmed in Java using Android Studio. Libraries like Volley will be used to handle the HTTP communication in the android app. For the server side we will be using the LAMP stack. The LAMP stack is the foundation for Linux hosted websites, it consists of Apache, MySQL and PHP software stack.

3.3.4 Communication Interfaces

The communication between the application and the web-server will be done using HTTP. The application will send a HTTP request to the web-server. The HTTP GET method will be used to request the data. The web-server returns a HTTP response containing JSON data related to the request. This JSON data is then parsed by the application.

In the absence of mobile network, we will be using MANET to communicate between the devices. The devices will form a decentralized peer-to-peer network. The MANET is formed using Wi-Fi Direct technology available on all modern android devices. This communication will be real-time to enable group chat among the peers.

3.4 Hardware and Software Requirements

3.4.1 Hardware Requirements

- A mobile phone with wireless connectivity is required to run the application. We use Wi-Fi Direct technology in aid in creating a MANET, so the mobile phone must also be equipped with this functionality.
- In order to store and provide data to the application, a reliable server is essential.

3.4.2 Software Requirements

- We have chosen to create this project for the Android operating system, due to it's extreme popularity and the user friendliness.
- To save records, we have chosen SQLlite on Android.
- Java and XML will be the main languages used in order to create the application, because that is how Android applications are made.
- Volley is a networking library was introduced to make networking calls much easier, faster without writing tons of code. By default all the volley network calls works asynchronously, which makes it ideal for our application
- LAMP is an archetypal model of web service stacks, named as an acronym of the names of its original four open-source components: the GNU/Linux operating system, the Apache HTTP Server, the MySQL relational database management system, and the PHP programming language. This will be used to create the web portal for the Rescue Coordinator to login and modify dathttps://www.overleaf.com/1914151899kqftthqkgsjza in the database.

3.5 Functional Requirements

This section includes the requirements that specify all the fundamental actions of the software system.

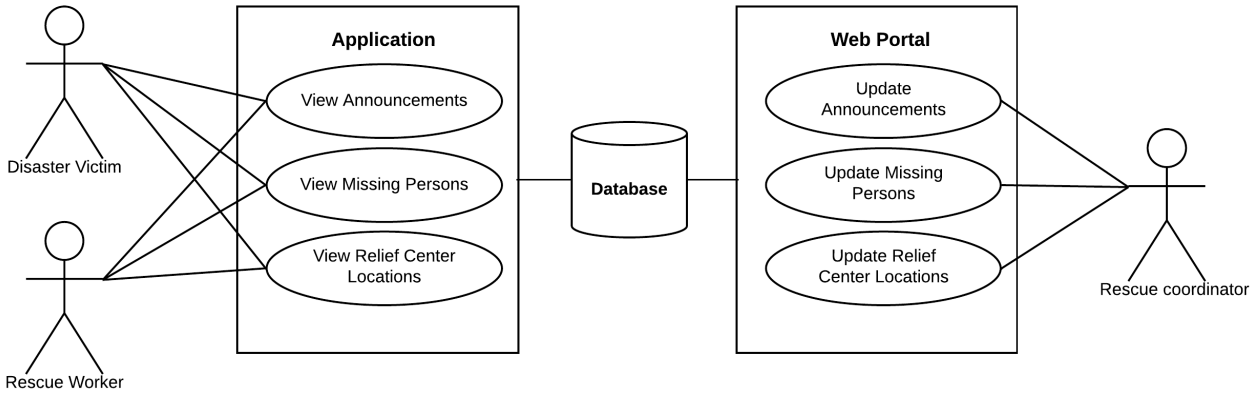

Figure 3.1: Use Case Diagram 1

3.5.1 View announcements

The users will be able to view the announcements in real time. These announcements may include any warnings or information regarding relief camps to be sent to the users in broadcast mode.

3.5.2 View missing persons database

The users will be able to view the missing person record in the database. In case a match is found, then display the persons profile. If a match is not found, then display an error message stating no such record exists.

3.5.3 Mark affected areas on the map

The users will be able to mark an affected area in their neighbourhood as dangerous. The GPS location is verified and the map is zoomed in to the user's location and the area is marked as dangerous.

3.5.4 Unmark affected areas on the map

The users will be able to unmark an affected area once it's declared safe. The GPS location is verified and the map is zoomed in to the user's location and the area is unmarked as dangerous.

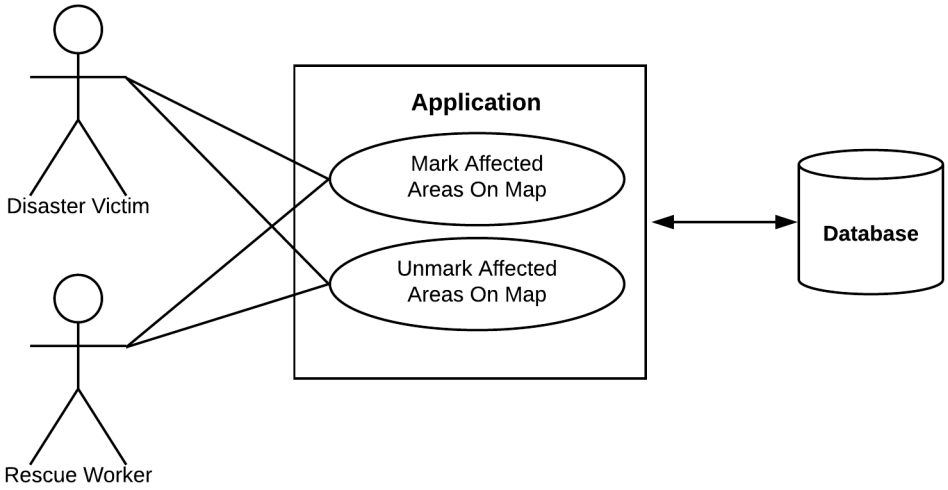

Figure 3.2: Use Case Diagram 2

3.5.5 Request for help along with geolocation

The users will be able to post a message request for help along with their geographical coordinates. The appropriate request message and current location of the user, obtained through GPS, is posted.

3.5.6 View the help requests and locations

The users will be able to view the requests posted in the help and support section. The user will be able to respond to requests. Once a request is accepted, it is marked as underway. After the request is complete, inform the rescue coordinator to mark the request as complete.

3.5.7 Text communication among the victims and rescue workers

In case of disruption of cellular signal, a basic method of text communication must be implemented for users to communicate, with the help of a MANET.

3.5.8 View Relief centre locations

The relief centres locations are pinned on the map which can be viewed by the users in the aftermath of a disaster.

3.5.9 Login for rescue coordinator

The rescue coordinators will be authenticated before accessing the web interface and given modification rights.

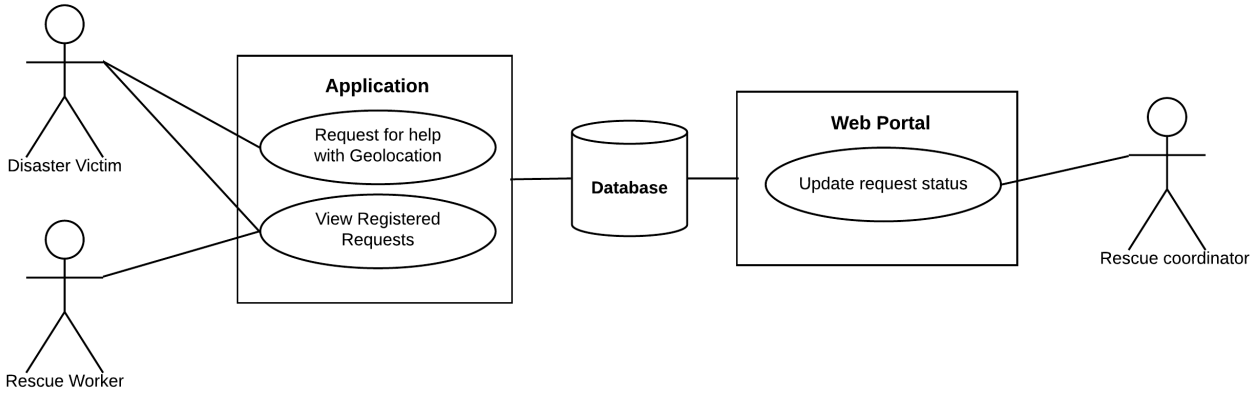

Figure 3.3: Use Case Diagram 3

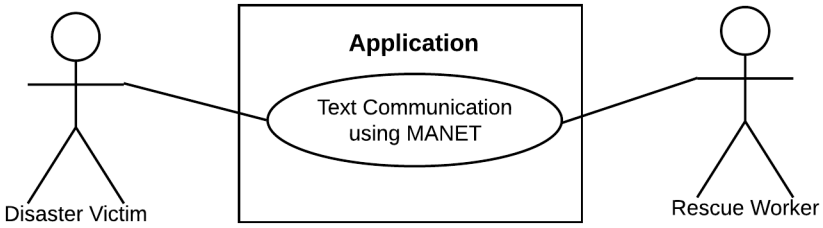

Figure 3.4: Use Case Diagram 4

3.5.10 Update the missing persons database

The rescue coordinator will be able to update the missing persons database with information received from the rescue workers and disaster victims.

3.5.11 Update announcements

The rescue coordinator will be able to add new announcements and remove or edit old announcements from the web interface.

3.5.12 Update relief center locations and donation sites

The rescue coordinator will be able to add new relief centre locations or donation sites from the web interface.

3.5.13 Update help requests status

The rescue coordinator will be able remove and mark help requests as completed. User notifies that the request is ongoing, coordinator updates status as under progress.

3.6 Non-functional Requirements

3.6.1 Performance Requirements

1. Updating the database will be done in real time, because delays as short as 20 minutes can lead to rescue workers utilizing resources incorrectly, which can cost time and lives.
2. If the system loses the connection to the Internet or to the GPS device or the system gets some strange input, the user will be informed.
3. The map data will be cached locally for reduced load-up time.

3.6.2 Security Requirements

1. Only the authenticated rescue coordinators will be allowed to make changes to the database via the web interface, so the login page will be reliable and not susceptible to attacks like SQL injection.
2. The website will use HTTPS protocol to encrypt the session.

3.6.3 Software Quality Attributes

Usability

Application will be easy to navigate through and use. All the basic features will be accessible readily. Only a minimum number of steps will be needed to post a request for help. The application will have fast response times in the user interface.

Reliability

The application will be reliable, as lives are at stake. The number of crashes must be kept to an absolute minimum. In case of crashes, data will not be lost. The server will be able to handle burst in high traffic in case of a disaster so that it doesn't crash.

Scalability

The system will be able to handle multiple users at a time. The new users will be incorporated into the network swiftly. Adding more nodes in the MANET will be simple, in order to allow transmission of information through a large area. Single points of failure will not hinder the entire application performance.

Maintainability

The system will be created in modules, for easier maintainability. Having boundaries and clear functionality among modules is always beneficial. Maintenance should be cost effective and easy.

3.7 Other Requirements

Reusability

The project is designed with reuse in mind, so it will be flexible enough to allow reuse in different countries, to aid in the mitigation of different kinds of disasters, both natural and man-made.

Chapter 4

System Design

4.1 System Architecture

In this section, we explain the detailed architecture of the proposed system. It also includes the architecture diagram and the explanation of the modules of the system.

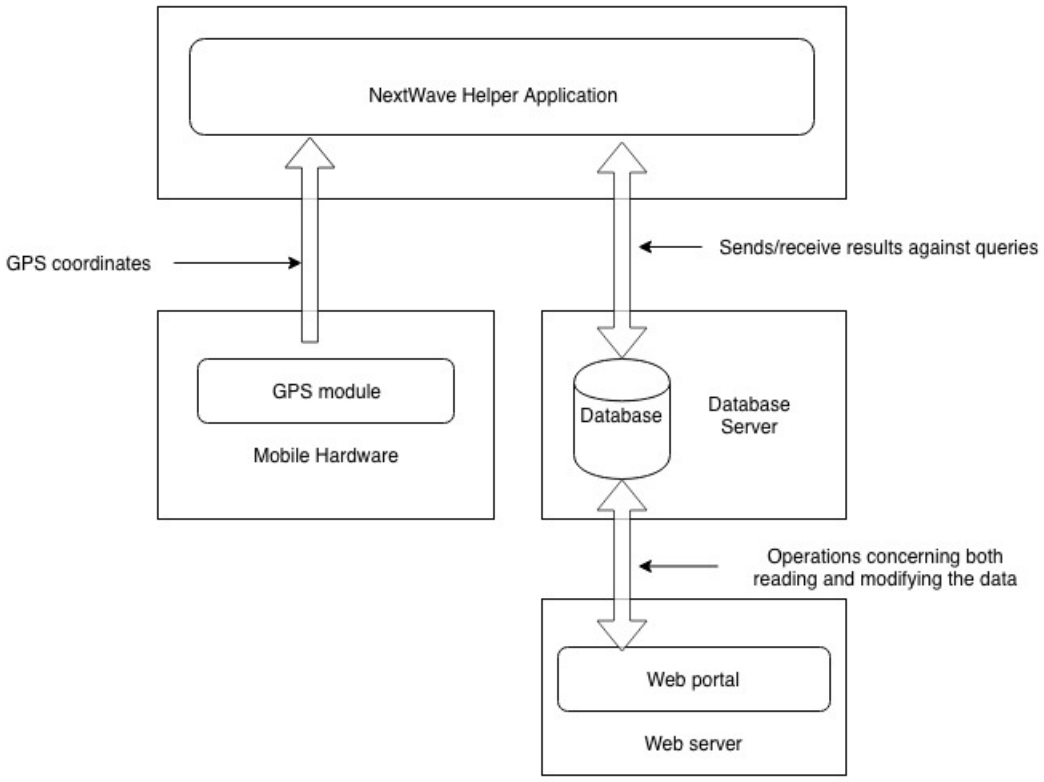

Figure 4.1: System Architecture

The system under consideration is based on a 3-tier peer-to-peer architecture (Presentation, Application and Data Layer).

4.1.1 GPS module

The physical location data is managed by the GPS module in the mobile phone. The mobile application communicates with the GPS module in order to get geographical information about where the user is located and the visual representation of it.

4.1.2 Database server

The NextWave Helper application interacts with the database server in order to get the information about affected areas and relief camps. It also stores missing person profiles and shows results against queries made by users.

4.1.3 Web portal

The communication between the database and the web portal consists of operation concerning both reading and modifying the application data such as announcements and relief centre locations.

4.2 Input Design

4.2.1 Use Case Diagram

Use case diagrams are considered to be used for high level requirement analysis of a system. So when the requirements of a system are analyzed the functionalities are captured in use cases. To draw a use case diagram we should have the following items identified:

- Functionalities to be represented as an use case.
- Actors.
- Relationships among the use cases and actors.

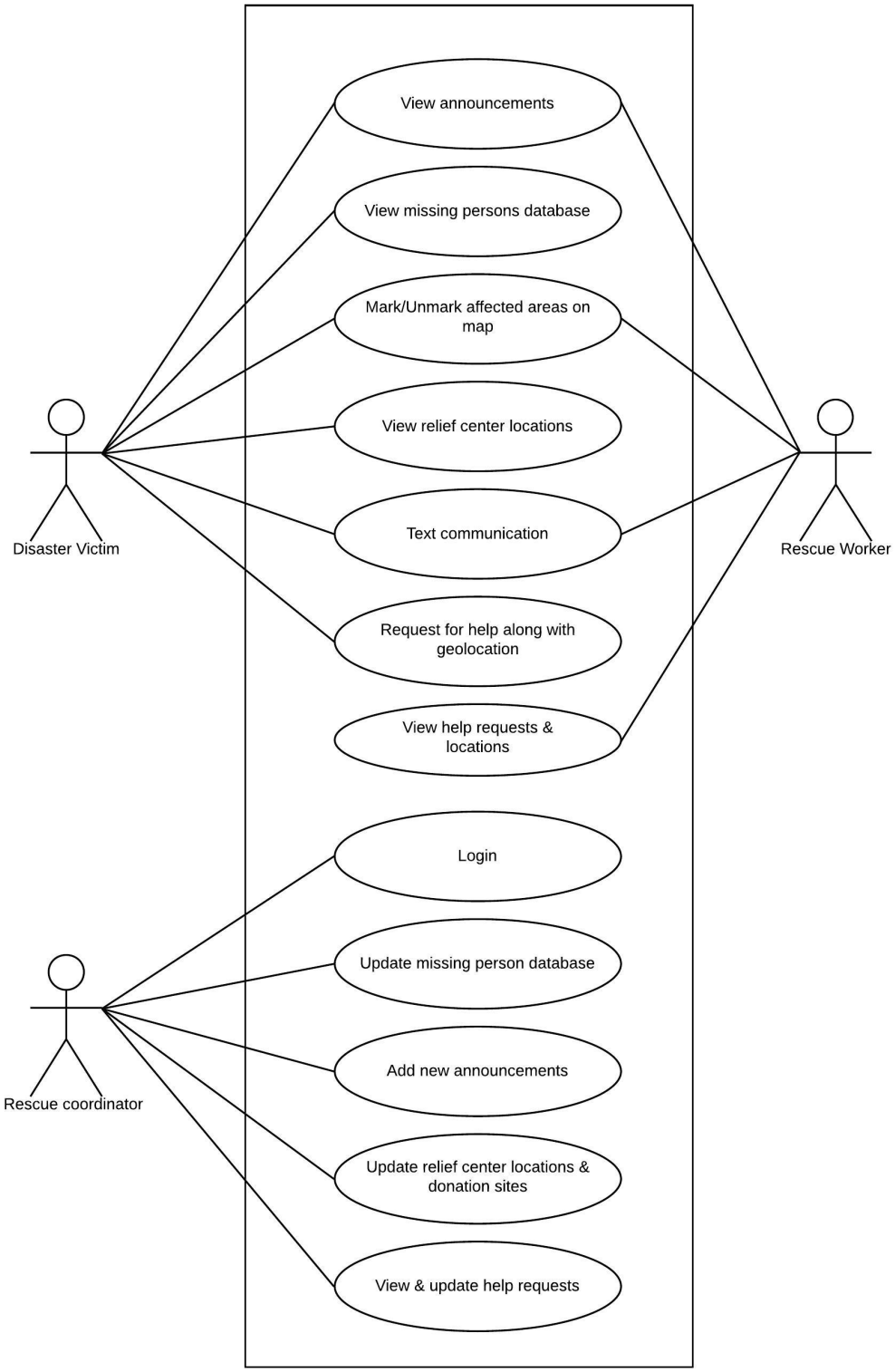

Figure 4.2: Use Case Diagram

4.2.2 View announcements

Description

The users will be able to view the announcements in real time. These announcements may include any warnings or information regarding relief camps to be sent to the users in broadcast mode.

Actors

Disaster victim, Rescue worker

Precondition

None

Main flow of events

1. The user clicks on the announcements button.
2. A list of already cached announcements are shown.
3. The application contacts the web server to grab new announcements.
4. New announcements are appended to the local cache and displayed.

Alternate flow of events

1. If there is no active internet connection then the list of locally cache announcements are displayed.

Post condition

The announcements are displayed on the screen.

4.2.3 View missing persons database

Description

The users will be able to view the missing person record in the database. In case a match is found, then display the persons profile. If a match is not found, then display an error message stating no such record exists.

Actors

Disaster victim, Rescue worker

Precondition

An updated and constantly monitored database

Main flow of events

1. The user types in the name of a potentially missing person in the search field.
2. This request is then checked against the other records in the database.
3. In case a match is found, then display the persons profile.
4. If a match is not found, then display an error message stating no such record exists.

Alternate flow of events

1. In case there is no internet connection then, an error message is shown.

Post condition

The potentially missing person's profile is displayed.

4.2.4 Mark affected areas on the map

Description

The users will be able to mark an affected area in their neighbourhood as dangerous. The GPS location is verified and the map is zoomed in to the user's location and the area is marked as dangerous.

Actors

Disaster victim, Rescue worker

Precondition

The user's device must have it's GPS turned on.

Main flow of events

1. The user opens the affected areas map.
2. The GPS location is verified and the map is zoomed in to the user's location.
3. If the user clicks on any roads near his/her GPS location, the area is marked as dangerous.

Alternate flow of events

1. If the user tries to mark any remote areas as affected, then an error message is shown.

Post condition

The road/street is marked as dangerous.

4.2.5 Unmark affected areas on the map

Description

The users will be able to unmark an affected area once it's declared safe. The GPS location is verified and the map is zoomed in to the user's location and the area is unmarked as dangerous.

Actors

Disaster victim, Rescue worker

Precondition

The user's device must have it's GPS turned on.

Main flow of events

1. The user opens the affected areas map.
2. A list of already cached announcements are shown.
3. The GPS location is verified and map is zoomed in to the user's location.
4. This sets the location back to safe.

Alternate flow of events

1. If the user tires to unmark any remote areas as safe, then an error message is shown

Post condition

The road/street is unmarked as dangerous.

4.2.6 Request for help along with geolocation

Description

The users will be able to post a message request for help along with their geographical coordinates. The appropriate request message and current location of the user, obtained through GPS, is posted.

Actors

Disaster victim

Precondition

The user's device must have it's GPS turned on and have an active cellular network connection.

Main flow of events

1. The user needs to open the help and support page.
2. Click on create new request.
3. Post the appropriate request message.
4. The current location of the user will be obtained through the GPS and coupled with the message.
5. The request is then registered and stored in a database.

Alternate flow of events

1. In the absence of a cellular network (or a very unstable network), the help request is not registered.
2. In case the GPS location is not available, the user can pin their location on the map.

Post condition

A request is registered and stored in database

4.2.7 View the help requests and locations

Description

The users will be able to view the requests posted in the help and support section. The user will be able to respond to requests. Once a request is accepted, it is marked as underway. After the request is complete, inform the rescue coordinator to mark the request as complete.

Actors

Rescue Worker, Rescue Coordinator

Precondition

The user must have an active cellular network connection.

Main flow of events

1. The user needs to open help and support section.
2. Click to view the active open requests.
3. The user can respond to requests that are active and open.
4. Once a request is accepted, it is marked as underway.
5. After the request is complete, inform the rescue coordinator to mark the request as complete.

Alternate flow of events

1. If the accepted request is not completed within a stipulated time, then mark it as open again.

Post condition

A request is viewed, responded to and completed.

4.2.8 Text communication among the victims and rescue workers

Description

In case of disruption of cellular signal, a basic method of text communication must be implemented for users to communicate, with the help of a MANET.

Actors

Disaster Victim, Rescue Worker

Precondition

Mobile devices have Wi-Fi Direct capabilities

Main flow of events

1. The user enables disaster mode in the application.
2. The user can now send text messages that are propagated to other users through the MANET.

Alternate flow of events

1. If there are no neighboring peers, an error message is shown.

Post condition

Text messages can be sent in order to communicate, even without cellular network

4.2.9 View Relief centre locations

Description

The relief centres locations are pinned on the map which can be viewed by the users in the aftermath of a disaster.

Actors

Disaster victim, Rescue worker

Precondition

An active cellular network connection to view maps.

Main flow of events

1. Select relief centres under maps section.
2. All institutions/areas that are relief centres are pinned on the map.

Alternate flow of events

1. In the absence of a cellular network connection, maps cannot be viewed.

Post condition

View the relief centre locations on the maps.

4.2.10 Login for rescue coordinator

Description

The rescue coordinators will be authenticated before accessing the web interface and given modification rights.

Actors

Rescue Coordinator

Precondition

None

Main flow of events

1. The user opens the web interface and enter the credentials.
2. If the credentials match the authorized users database, the home page is displayed.

Alternate flow of events

1. If the user is not authenticated, then the login fails and an error message is shown.

Post condition

The homepage is displayed.

4.2.11 Update the missing persons database

Description

The rescue coordinator will be able to update the missing persons database with information received from the rescue workers and disaster victims.

Actors

Rescue Coordinator

Precondition

The user must be logged in as rescue coordinator.

Main flow of events

1. The rescue coordinator receives information from a user, typically a rescue worker.
2. The rescue coordinator opens the database interface and adds the new information to the database.

Alternate flow of events

1. The rescue coordinator receives information to update from a user, typically a rescue worker.
2. The rescue coordinator updates the information of the person in the database(such as marking a missing person as found).

Post condition

The database is updated with new/correct information.

4.2.12 Update announcements

Description

The rescue coordinator will be able to add new announcements and remove or edit old announcements from the web interface.

Actors

Rescue Coordinator

Precondition

The user must be logged in as rescue coordinator.

Main flow of events

1. If the user can click on the add button inside the announcements page to add new announcements.
2. If the user wants to edit any announcement, then the user can click on update button displayed next to each announcement on the list.

Alternate flow of events

1. If the user wants to remove any announcement, the user can click on the remove button next to an announcement.

Post condition

The announcements list gets updated.

4.2.13 Update relief center locations and donation sites

Description

The rescue coordinator will be able to add new relief centre locations or donation sites from the web interface.

Actors

Rescue Coordinator

Precondition

The user must be logged in as rescue coordinator.

Main flow of events

1. The user receives information about new centre locations and potential donation sites from other users.
2. This information is then verified.
3. If it's legitimate, add and update the new relief centres/donation sites to the list of existing ones.

Alternate flow of events

1. In case the information cannot be verified, the list is not updated

Post condition

The updated list depicts the new locations and donation sites.

4.2.14 Update help requests status

Description

The rescue coordinator must be able remove and mark help requests as completed.

Actors

Rescue Coordinator

Precondition

The rescue coordinator has received information about the request being completed.

Main flow of events

1. The coordinator checks the current status of the request.
2. In case the user notifies that the request is ongoing, coordinator updates status as under progress.
3. In case the request has been completed by the user, coordinator updates status as closed.

Alternate flow of events

1. If a request is not attended to within the stipulated time, update the status as active and open again.

Post condition

The status of help requests are updated.

4.3 Database Design

We would have a single database with multiple entities, which are as follows:

1. **Admin Users:** Store the details for the login credentials of the rescue coordinator.
2. **Announcements:** Store the announcements and the level of importance.
3. **Relief Center Locations:** This table will store the data related to relief center camps and collection centers. This data will used to display on the map.
4. **Affected Locations:** Contains the data regarding the areas which are affected, this information will be used to display as dangerous on the map.
5. **Help Requests:** Stores the help requests data along with status of the request.
6. **Missing Persons:** A table for storing the information on lost/missing persons as well as those who are found.

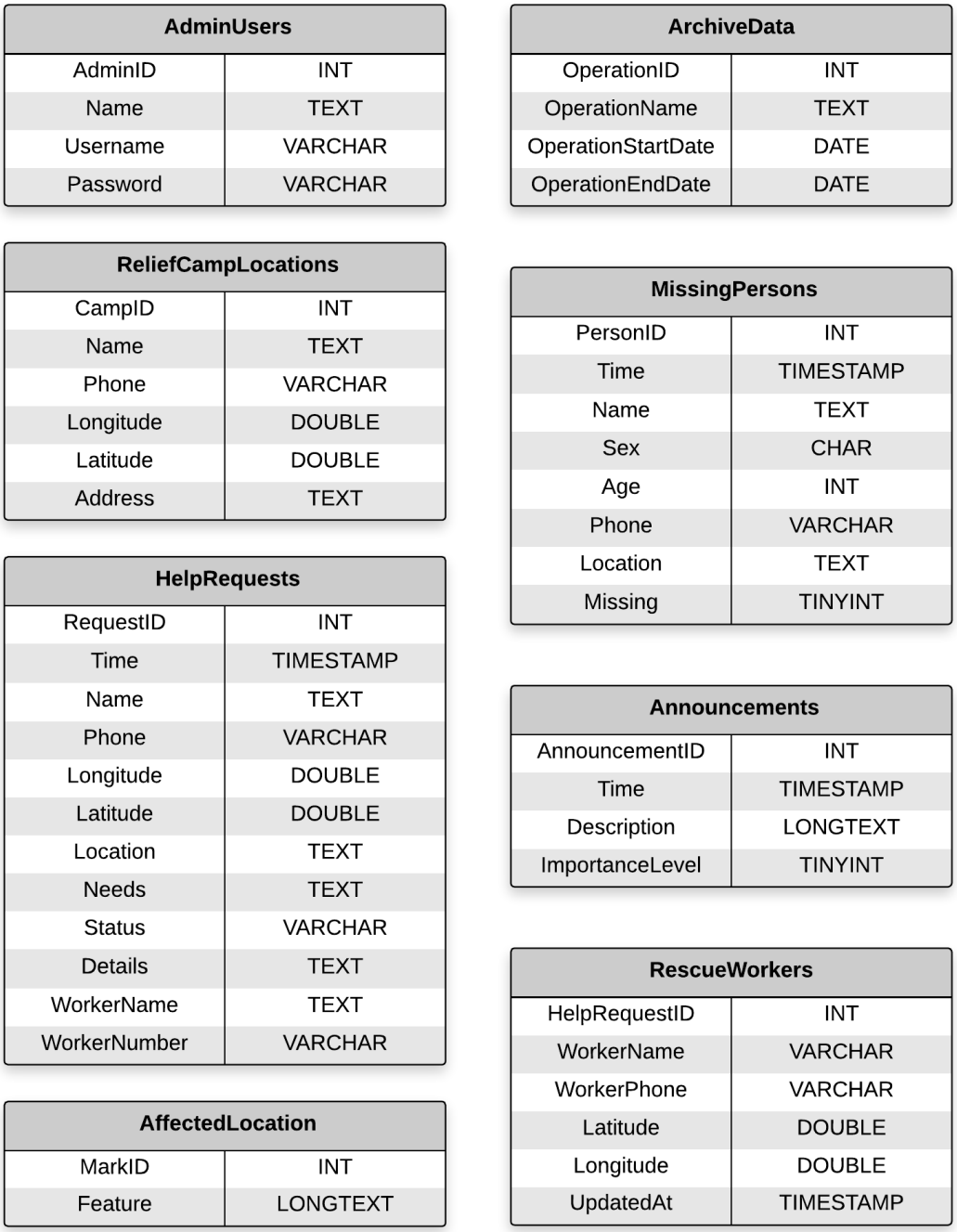

Figure 4.3: Database Schema

4.4 Class Diagram

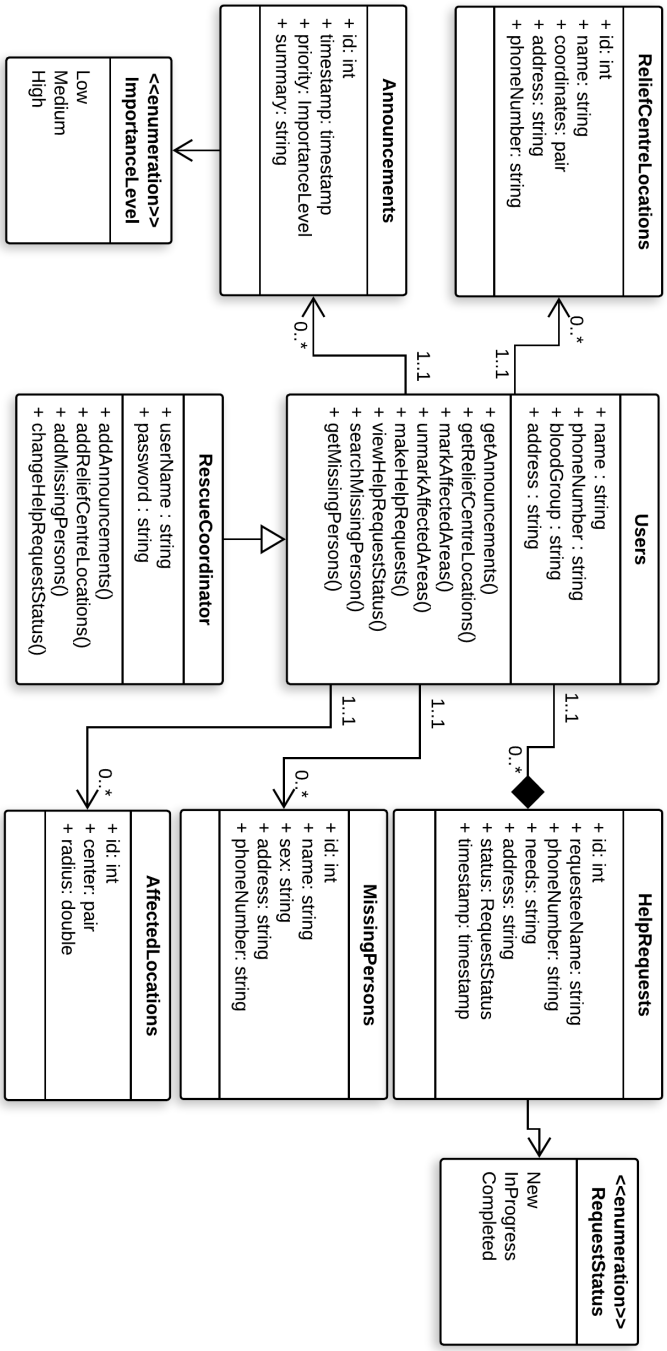

Figure 4.4: Class Diagram

4.5 Activity diagram

4.5.1 View announcements

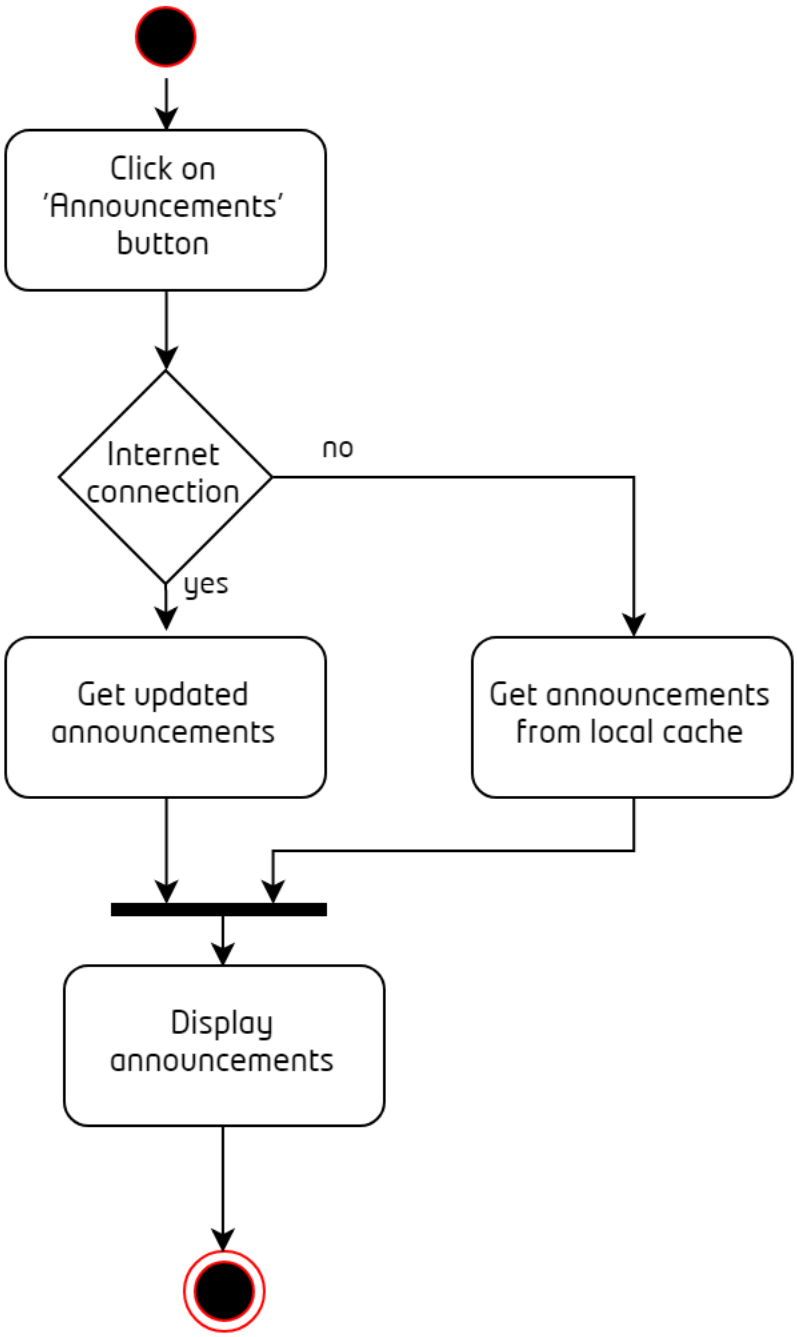

Figure 4.5: View announcements

4.5.2 View missing persons database

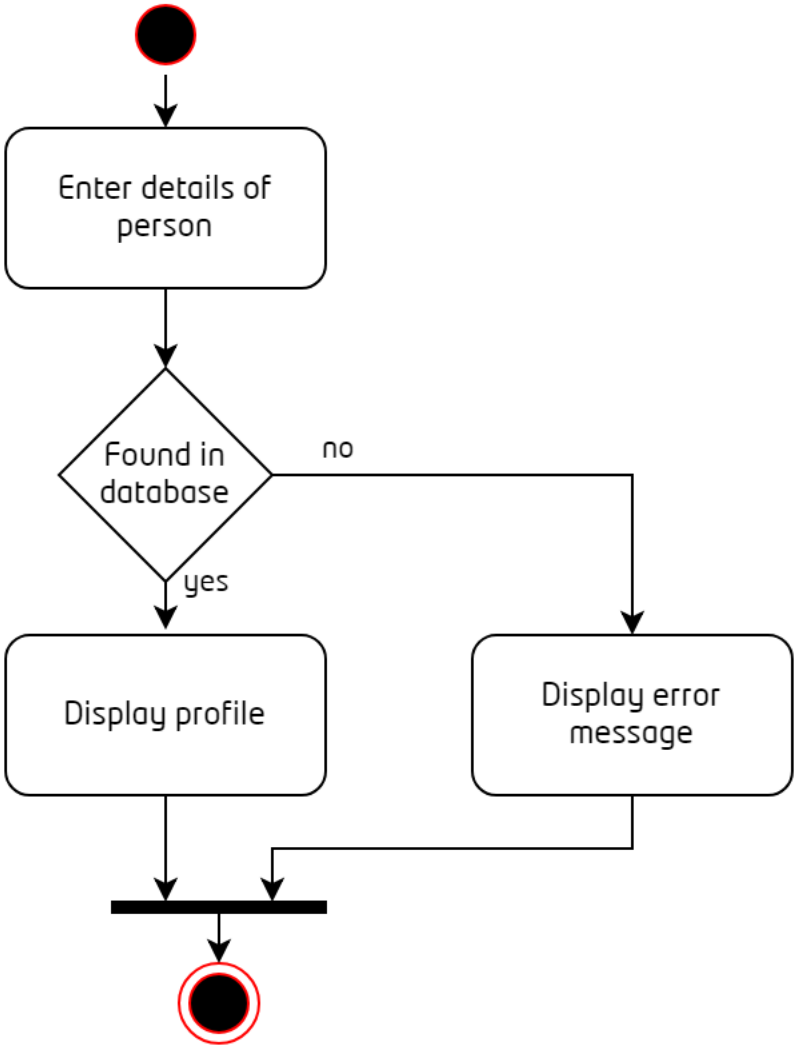

Figure 4.6: View missing person database

4.5.3 Mark affected areas on the map

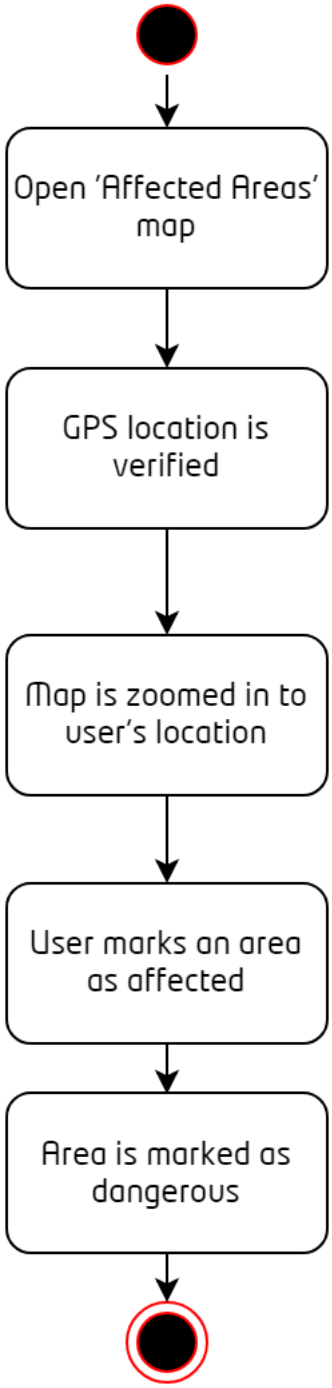

Figure 4.7: Mark affected areas on the map

4.5.4 Unmark affected areas on the map

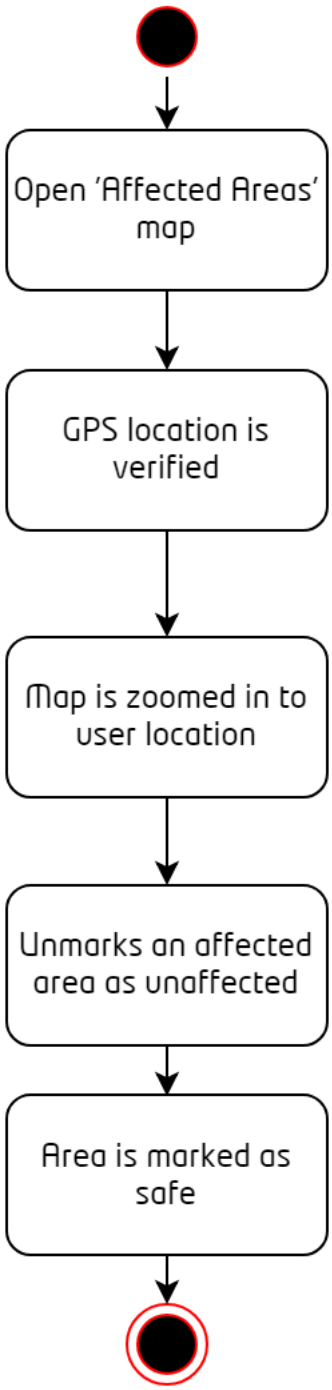

Figure 4.8: Unmark affected areas on the map

4.5.5 Request for help along with geolocation

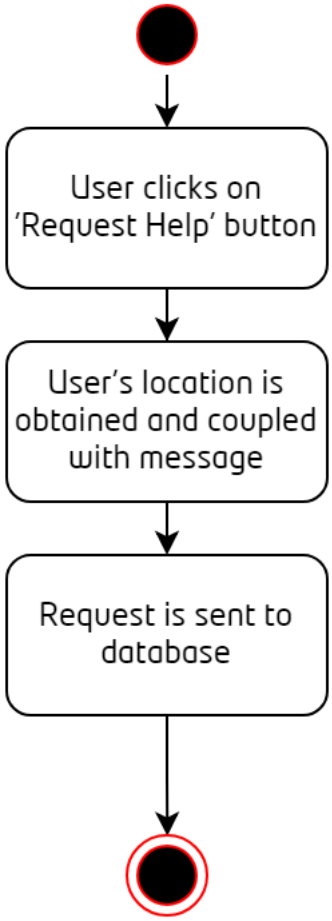

Figure 4.9: Request for help along with geolocation

4.5.6 View the help requests and locations

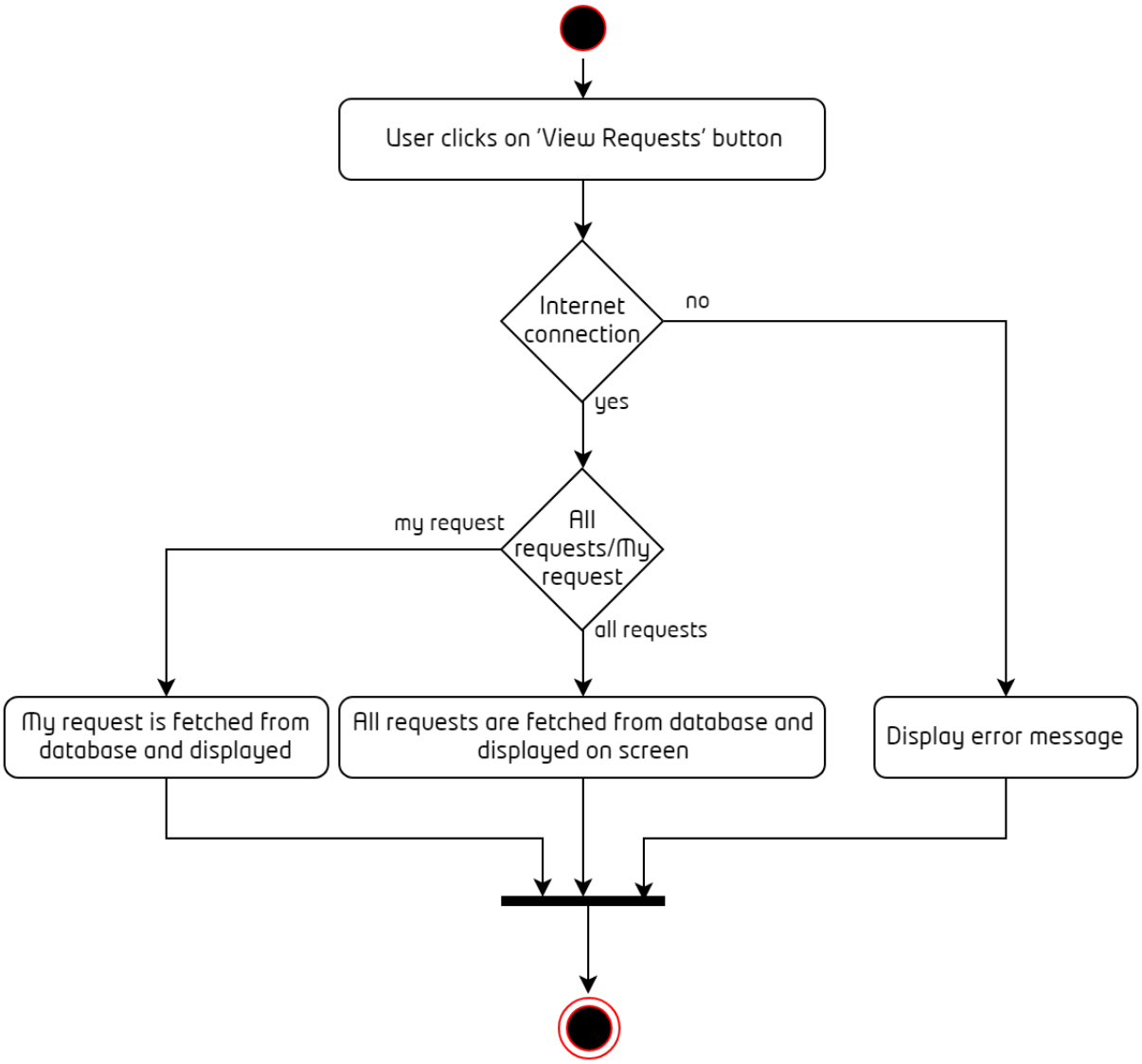

Figure 4.10: View the help requests and locations

4.5.7 Text communication among the victims and rescue workers

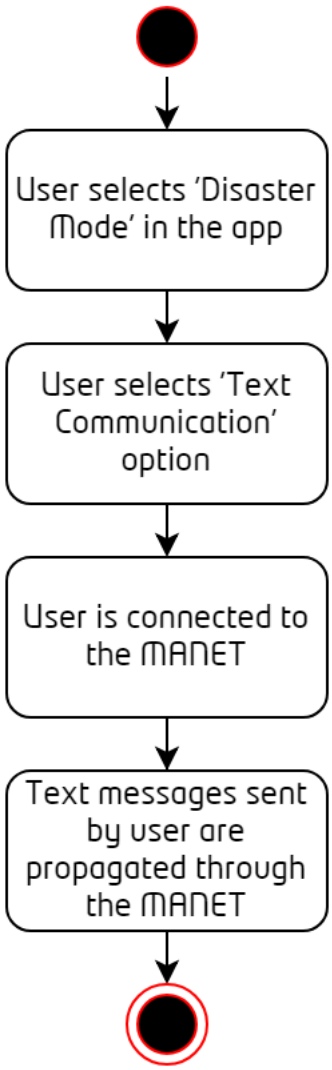

Figure 4.11: Text communication among the victims and rescue workers

4.5.8 View Relief centre locations

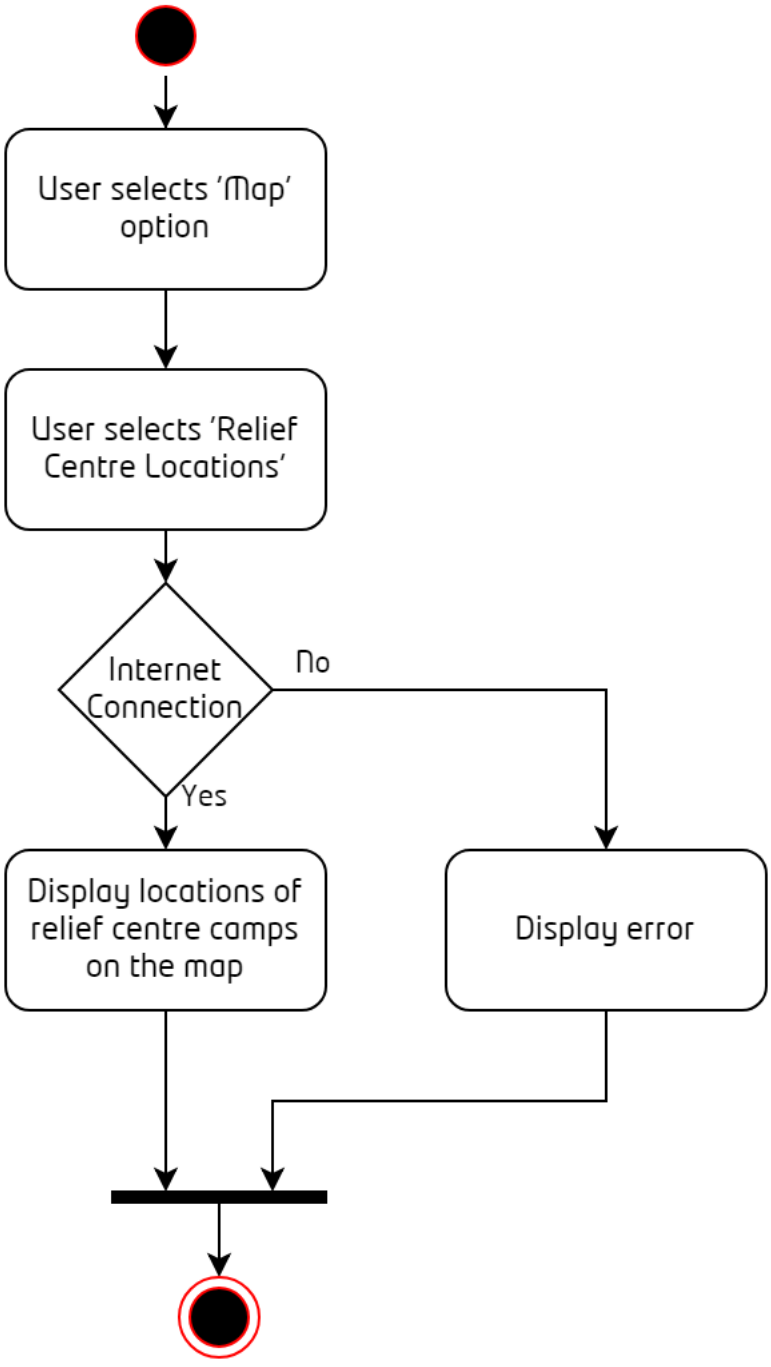

Figure 4.12: View Relief centre locations

4.5.9 Login for rescue coordinator

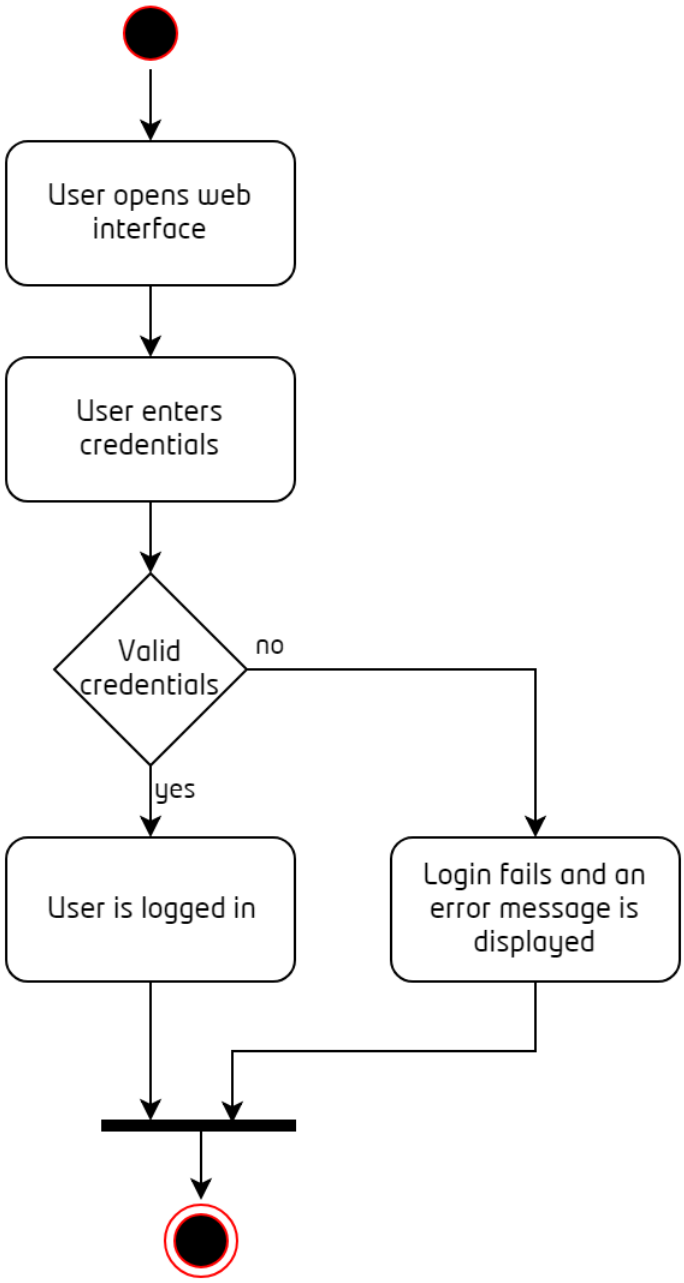

Figure 4.13: Login for rescue coordinator

4.5.10 Update the missing persons database

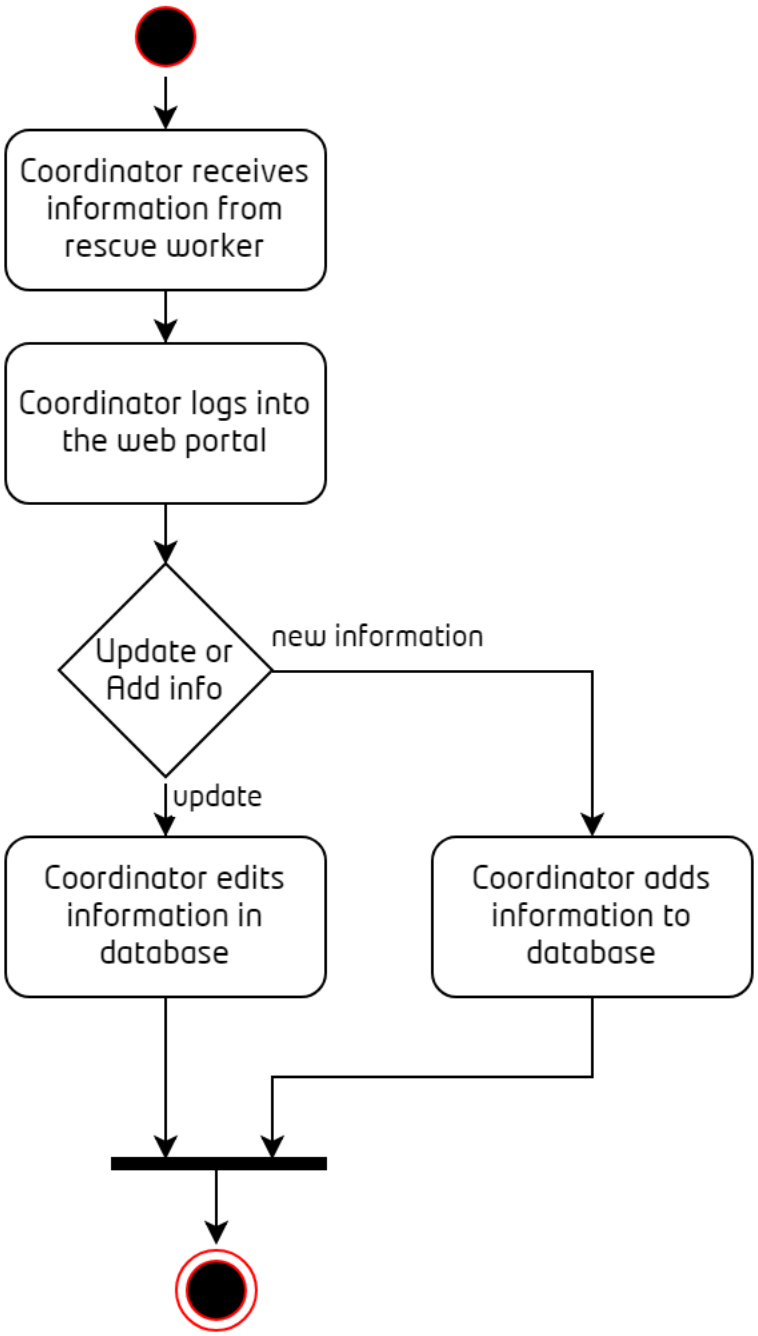

Figure 4.14: Update the missing persons database

4.5.11 Update announcements

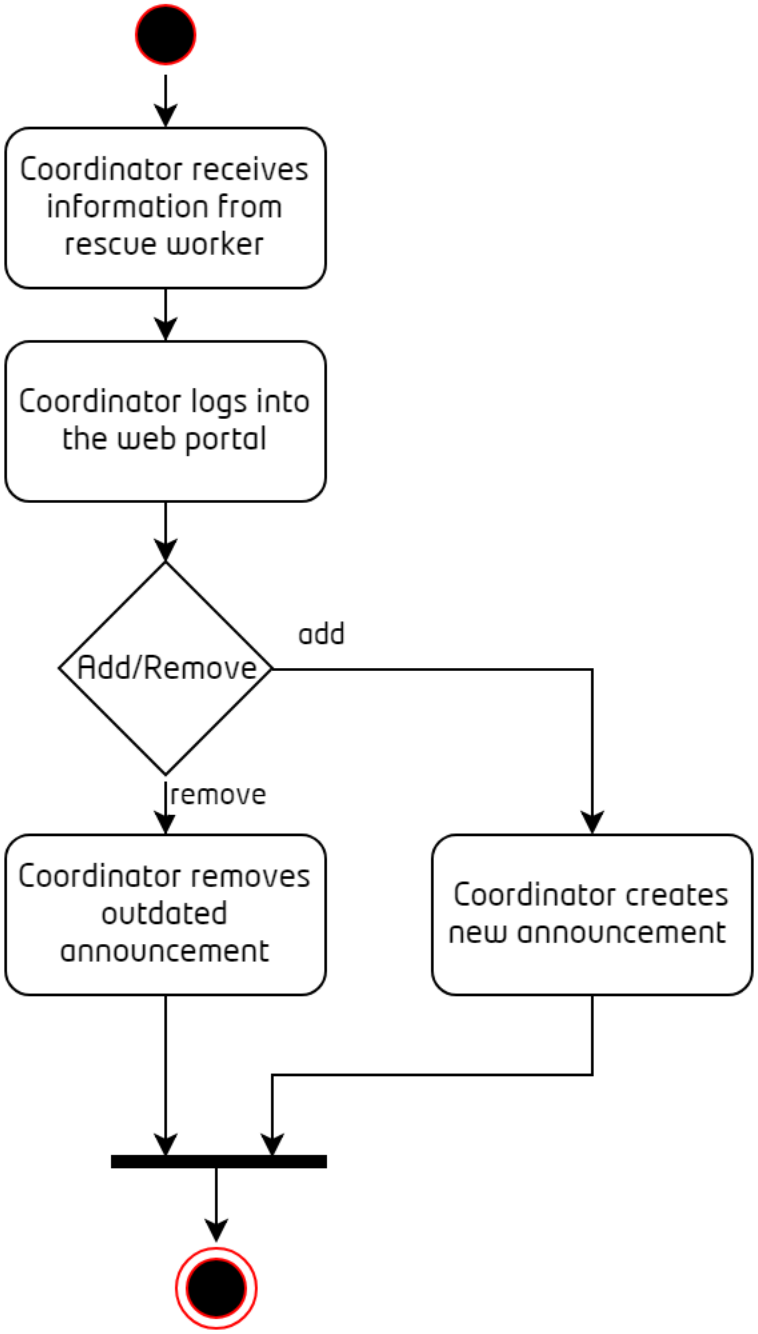

Figure 4.15: Update announcements

4.5.12 Update relief center locations and donation sites

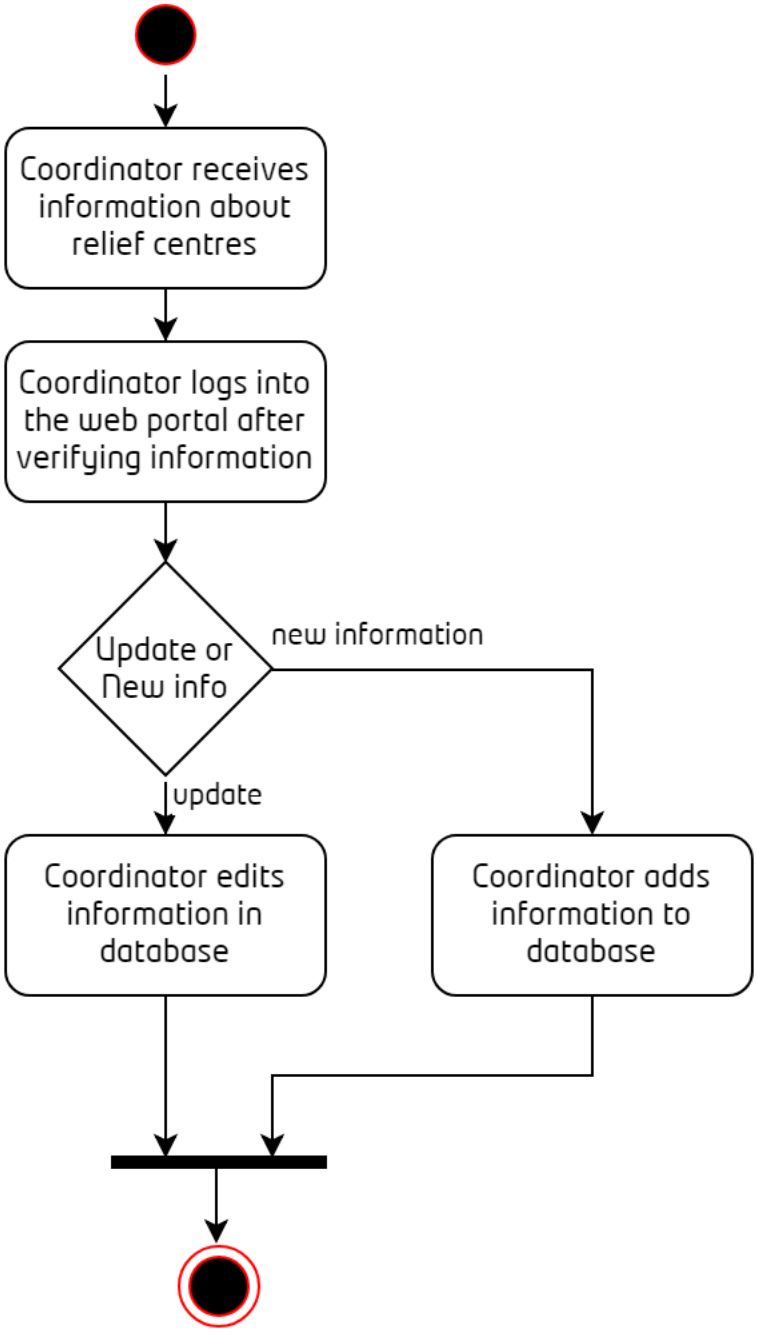

Figure 4.16: Update relief center locations and donation sites

4.5.13 Update help requests status

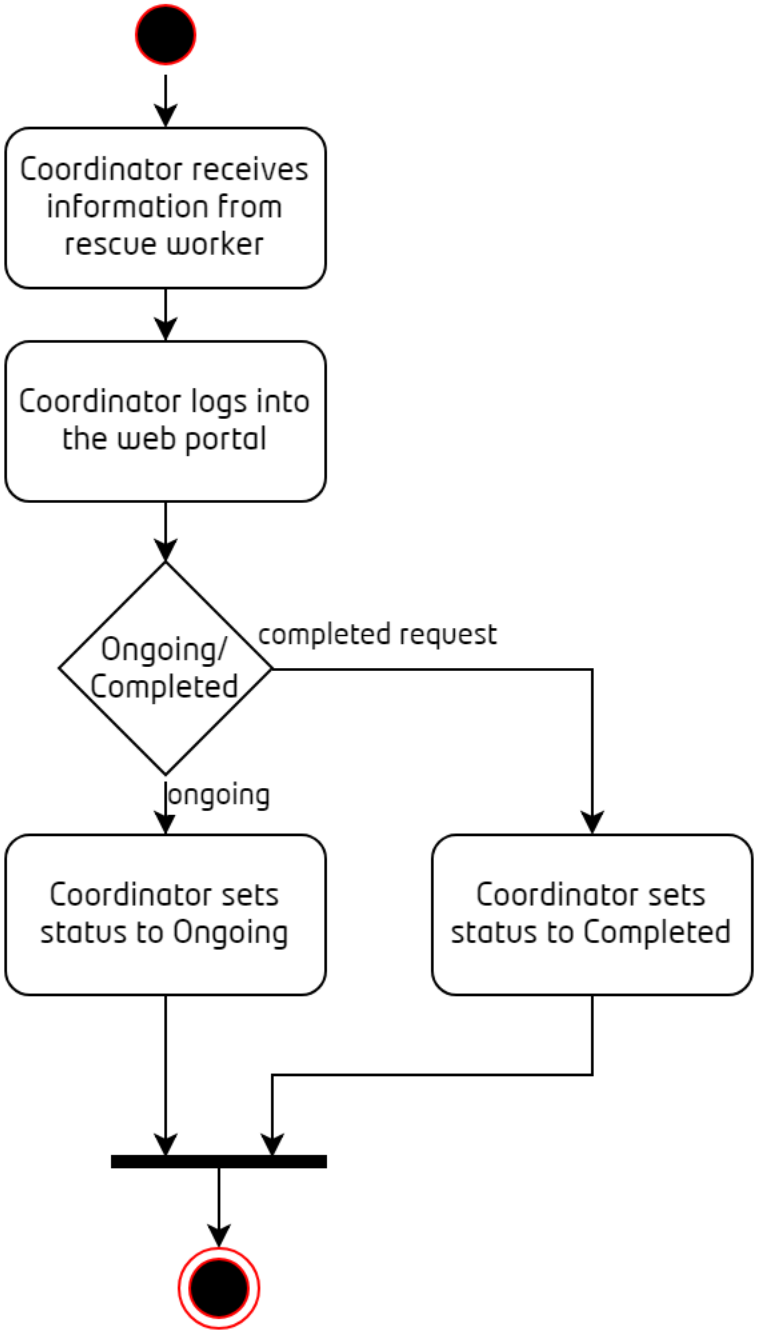

Figure 4.17: Update help requests status

4.6 Libraries and Packages Used

4.6.1 Retrofit

Retrofit is a HTTP client for Android and Java by Square, Inc. It turns HTTP APIs to Java interfaces.

Retrofit was used for communicating with our API server. It uses JSON format to transmit/receive data.

URL: `http://square.github.io/retrofit/`

4.6.2 Volley

Volley is a HTTP library developed by Google. This library is used to transmit data over the network. It makes networking faster and easier for Apps. The volley library has the features like automatic scheduling of network request, multiple concurrent connections, request prioritization, cancel/block a request, easier management of UI with data fetched asynchronously from the network and also offers easier customization.

Volley was used in the Android application to transmit/receive GeoJSON data from the webserver. GeoJSON is used for representing geographical features for affected locations feature.

URL: `https://github.com/google/volley`

4.6.3 Mapbox Maps Android SDK

The Mapbox Maps SDK for Android is an open source toolset for displaying maps inside of your Android application. It takes stylesheets that conform to the Mapbox Style Specification, applies them to vector tiles that conform to the Mapbox Vector Tile Specification, and renders them using OpenGL.

The Mapbox Maps SDK was used in the andorid application for embedded maps. The annotations plugin was used for adding markers.

URL: `https://github.com/mapbox/mapbox-gl-native/tree/master/platform/android`

4.6.4 Dagger

Dagger is a fully static, compile-time dependency injection framework for both Java and Android. It is an adaptation of an earlier version created by Square and now maintained by Google.

Dagger was used to inject singletons into fragments and actvities on Android. This helped to improve the code efficiency by handling creation of singleton instances, which meant less initialization code.

URL: `https://google.github.io/dagger/`

4.6.5 Gson

Gson is a Java library that can be used to convert Java Objects into their JSON representation. It can also be used to convert a JSON string to an equivalent Java object.

We used Gson in the Android app to parse the JSON string received from the API server and convert it to Java class objects for handling it in code. It also did the reverse conversion when we wanted to send data to the API server in form of JSON string.

URL: `https://github.com/google/gson`

4.6.6 Room Persistence Library

The Room persistence library provides an abstraction layer over SQLite to allow for more robust database access while harnessing the full power of SQLite. It is part of the Android Jetpack libraries.

The library helps to create a cache of the app's data on a device. This cache, which serves as the app's single source of truth, allows users to view a consistent copy of key information within the app, regardless of whether users have an internet connection.

We used the Room Persistance Library for storing the data locally on the device in a database. This allowed the app to display information even if there is no internet connection.

URL: `https://developer.android.com/topic/libraries/architecture/room`

4.6.7 Nearby Connection API

Nearby Connections is a peer-to-peer networking API that allows apps to easily discover, connect to, and exchange data with nearby devices in real-time, regardless of network connectivity.

Nearby Connections enables advertising, discovery, and connections between nearby devices in a fully-offline peer-to-peer manner. Connections between devices are high-bandwidth, low-latency, and fully encrypted to enable fast, secure data transfers. Under the hood, the API uses a combination of Bluetooth, BLE, and Wifi hotspots.

We used the Nearby Connections API to connect devices in peer-to-peer manner in disaster mode of the app. Using the API, messages and location updates can be shared.

URL: `https://developers.google.com/nearby/connections/overview`

4.6.8 Lumen

Laravel Lumen is a fast PHP micro-framework for building web applications. Lumen attempts to help with development by easing common tasks used in the majority of web projects, such as routing, database abstraction, queueing, and caching.

Lumen framework was used to develop the API server.

URL: `https://github.com/laravel/lumen-framework`

4.7 Module Description

4.7.1 Web Interface

The web interface of the application is built with HTML, CSS, Javascript and PHP. The web interface is used by the rescue coordinator to:

- Assign help requests to rescue workers based on which rescue worker is closest to the request location.
- Update status of help requests and missing persons
- Add new relief centres and announcements
- View and analyze the analytics, and work out areas that require more manpower or resources. Analytics should also be stored for posterity.

4.7.2 API

The API is built using Lumen - a fast PHP micro-framework used to build microservices and APIs. The API is an essential module that handles communication between the application and the database. Whenever a user desires to add a new help request or report a new missing person, it is done by sending a POST request to the API. In addition, each user obtains information from the database through GET requests.

4.7.3 Android Application - Normal mode

The normal mode of the application is used when cell connectivity is available. In the normal mode of the application, the user can:

- View announcements
- View missing persons and help requests
- Add missing persons and help requests
- View relief camps
- View areas affected by the disaster

4.7.4 Android Application - Disaster mode

Disaster mode is meant to be used when normal modes of communication(such as phone and network signal) are unavailable. It allows users to chat and share locations with nearby users, using Bluetooth LE, a technology that provides considerably reduced power consumption and cost while maintaining a similar communication range as compared to standard Bluetooth.

Chapter 5

Data Flow Diagram

A data flow diagram (DFD) is a graphical representation of the flow of data through an information system, modelling its process aspects. Often they are viewed as a preliminary step used to create an overview of the system which can be later elaborated. DFDs can also be used for the visualization of data processing.

5.1 Level 0 DFD

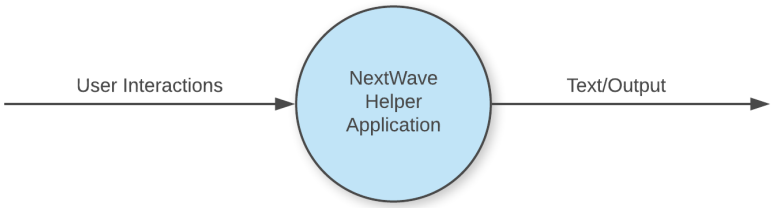

Figure 5.1: DFD Level 0

5.2 Level 1 DFD

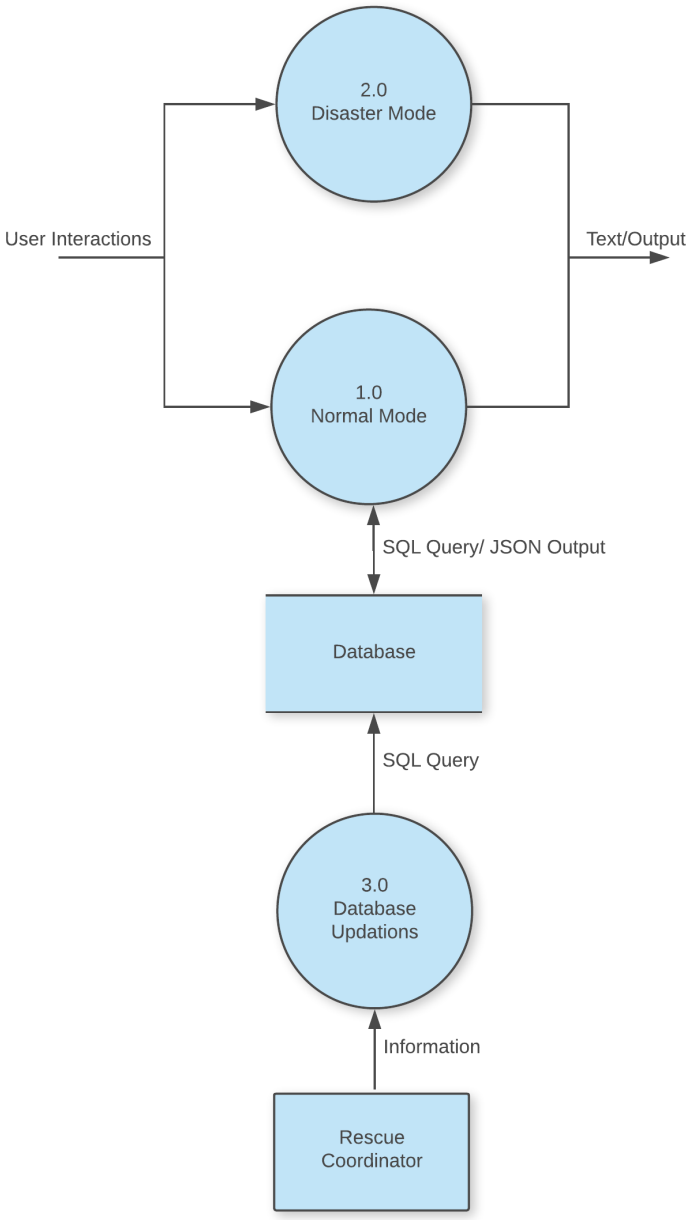

Figure 5.2: DFD Level 1

5.3 Level 2 DFD

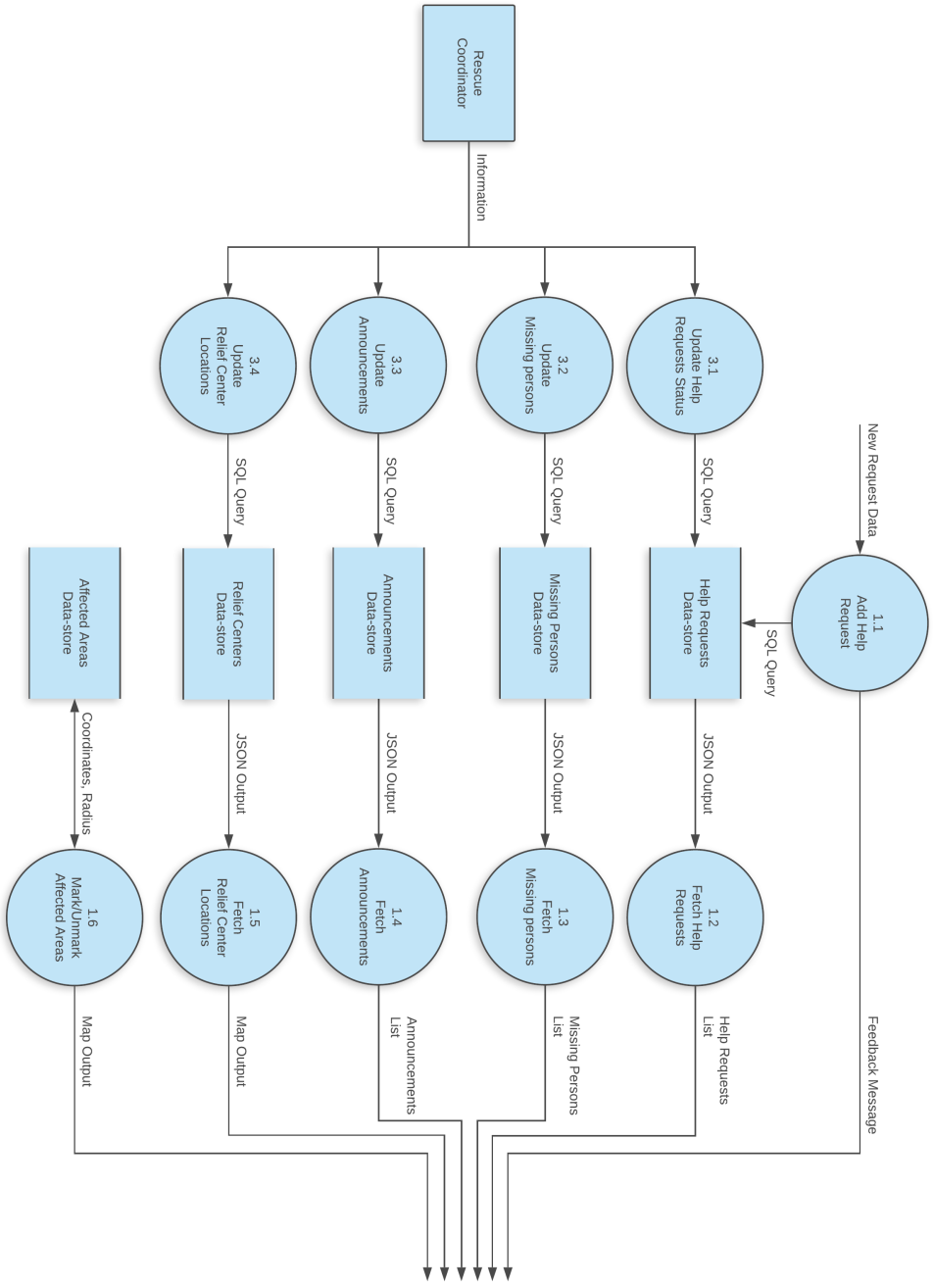

Figure 5.3: DFD Level 2 : Database updations and Normal Mode Process

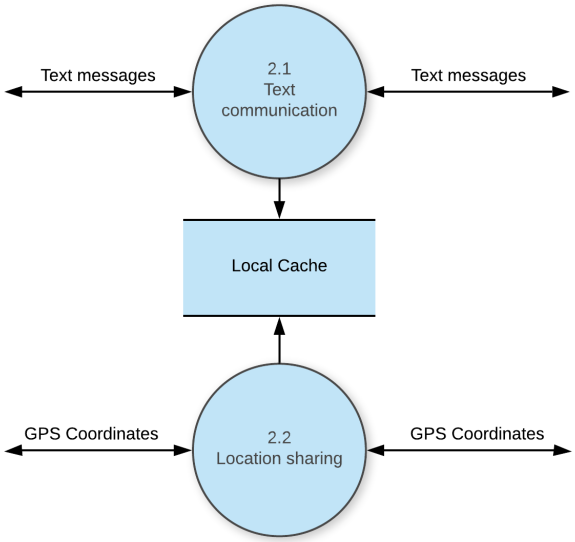

Figure 5.4: DFD Level 2 : Disaster Mode Process

Chapter 6

Implementation

6.1 Algorithms

6.1.1 Setup P2P Connection

The following algorithm highlights the basic steps involved in setting up the P2P connection between two Android devices using the Nearby Connections API.

Algorithm 1 Setup P2P Connection

```
1: Start
2: startAdvertising()                          ▷ starts advertising service to nearby devices
3: startDiscovering()                          ▷ starts discovering for nearby devices
4: if onEndpointFound() called then
5:     stopDiscovering()                       ▷ stops discovering nearby devices
6:     requestConnection()                     ▷ requests connection to the found endpoint
7:     if acceptConnection() then
8:         onConnectionResult() callback       ▷ This is called on both devices
9:         Connection established
10:    else if rejectConnection() then
11:        startDiscovering()                  ▷ starts discovering for any other devices
12: End
```

6.1.2 P2P MANET reactive routing

The following algorithm explains the steps in forwarding a message from source to destination node in a P2P MANET, using reactive routing.

Algorithm 2 AODV routing algorithm

```
1: Start
2: sender broadcasts RREQ to neighbours                ▷ RREQ: Route Request
3: for each node on receiving RREQ do
4:     if (currNode == destination) then
5:         update currNode's route table               ▷ seq_num, dest, next, hop
6:         send RREP to the source                     ▷ RREP: Route Reply
7:         for each node on receiving RREP do
8:             if (node == source) then
9:                 update node's route table
10:                start data transmission on discovered route
11:                End
12:            else if (node != source) then
13:                update node's route table
14:                forward RREQ
15:    else if (currNode != destination) then
16:        update currNode's route table
17:        forward RREQ
18: End
```

6.2 Development Tools

6.2.1 Git

Git is a version control system (VCS) for tracking changes in computer files and coordinating work on those files among multiple people. It is primarily used for software development, but it can be used to keep track of changes in any file. As a distributed revision control system it is aimed at speed, data integrity and support for distributed, non-linear work flow.

We had three Git repositories hosted on Gitlab. The three repositories were:

- Web Application
- Android Application
- API Server

We followed the 'Gitflow' workflow which meant we had two long running branches - 'Master' and 'Develop'. 'Master' branch was the live branch and had stable code. The 'Develop' branch was used to add features and bug fixing. Once everything was thoroughly checked, the changes from 'Develop' branch was merged to the 'Master' branch.

6.2.2 XAMPP/LAMP

XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages.

The stack offered by XAMPP was used as the platform for the web application server and the API server. For the web server, PHP was used for connecting to the MariaDB database and retrieving data by executing SQL queries. For the API server, Lumen framework was used to create endpoints that served JSON data.

6.2.3 Visual Studio Code

Visual Studio Code (VS Code) is a source-code editor developed by Microsoft. It includes support for debugging, embedded Git control, syntax highlighting, intelligent code completion, snippets, and code refactoring.

VS Code was used as the primary text editor for Web server and API server. The inbuilt support for embedded Git control helped us to get familiar with the workings of Git without dealing with CLI commands. The auto complete feature reduced the difficulties while developing the applications.

6.2.4 Android Studio

Android Studio is the official integrated development environment for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development.

To support application development within the Android operating system, Android Studio uses a Gradle-based build system, emulator, code templates, and Git integration. Android Studio uses an Instant Push feature to push code and resource changes to a running application. A code editor assists the developer with writing code and offering code completion, refraction, and analysis.

We used Android Studio to develop the NextWave Helper App. The inbuilt support for debugging and code linting helps fix bugs and identify errors efficiently. The support for Git helped us commit and push right from Android Studio, It also has a visual merge conflict resolver that helped us merge branches with ease.

Android Studio also managed the library dependencies and notified whenever a newer version of the library was available. The inbuilt user interface designer helps to visualize the UI normally written in XML as rendered image.

6.2.5 Postman

Postman is an API(application programming interface) development tool which helps to build, test and modify APIs. It has the ability to make various types of HTTP requests(GET, POST, PUT, PATCH) and provides a nice GUI for constructing requests and reading responses. It is an extremely useful tool during the development and testing of APIs.

Postman was used to test the API endpoints during development.

Chapter 7

Testing

7.1 Testing Methodologies

Software Testing Methodology is defined as strategies and testing types used to certify that the Application Under Test meets user expectations. Test Methodologies include functional and non-functional testing to validate the Application Under Test. Each testing methodology has a defined test objective, test strategy and deliverables. We performed the following testing: Unit testing, Integration testing, System testing.

7.2 Unit Testing

Unit testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. We considered 3 units for this process:

- Website - Tested if all the data is displayed properly from the database. Also tested if adding/updating works for all the features. The API endpoints were tested and it was confirmed that the authorization header is required for every call to the API.
- Android Application (Normal Mode) - Tested individual features of the app and made sure all the information is correctly displayed. Tested the application on different android versions and screen sizes.
- Android Application (Disaster Mode) - Tested group communication with more than 5 devices.

7.3 Integration Testing

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. Integration testing is conducted to evaluate the compliance of a system or component with specified functional requirements. The modules were integrated one by one and the partially integrated system was tested.

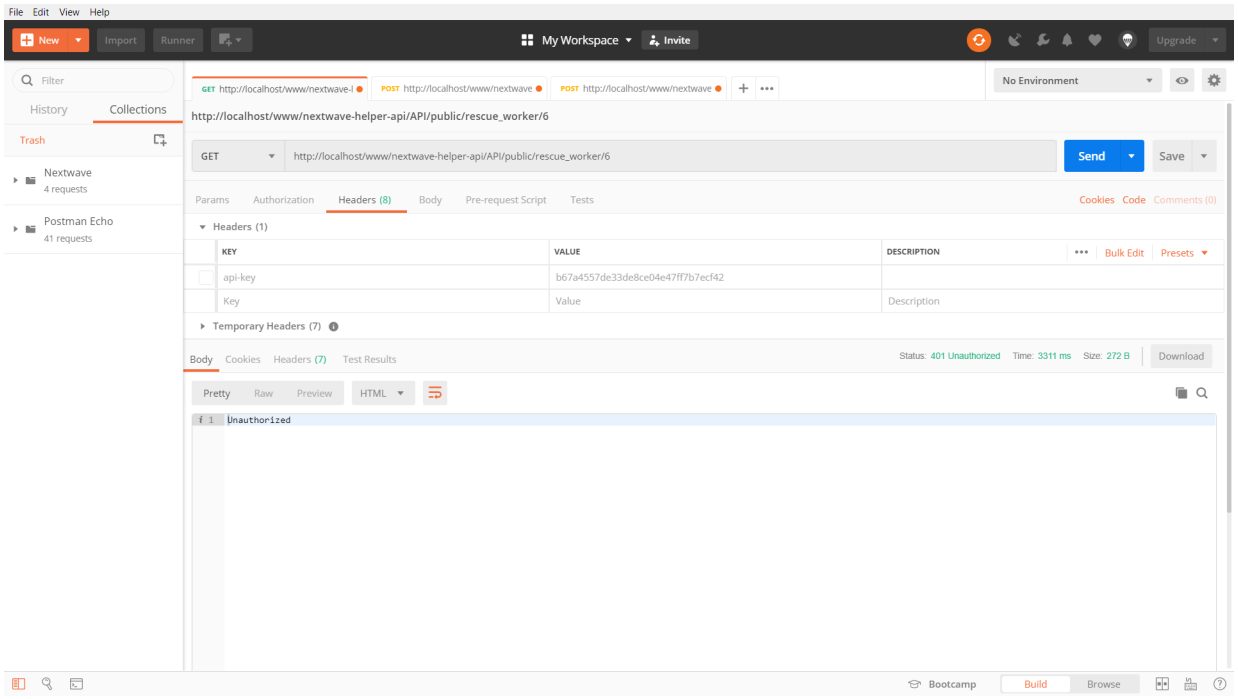

Figure 7.1: Trying to call the API without the authorization header

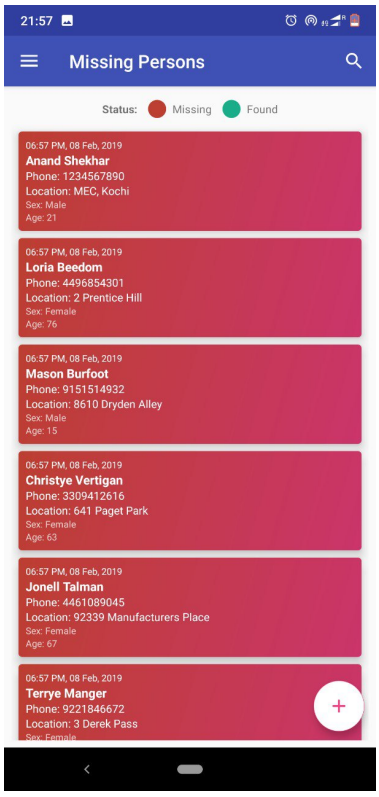

Figure 7.2: The Android Application getting data from the API

7.4 System Testing

System testing is testing conducted on a complete integrated system to evaluate the system's compliance with its specified requirements. System testing takes, as its input, all of the integrated components that have passed integration testing. The purpose of integration testing is to detect any inconsistencies between the units that are integrated together.

All modules were integrated at the end of integration testing and the entire system was tested.

Chapter 8

Graphical User Interface

8.1 GUI Overview

This project consists of 2 components - Android Application and Website.

8.2 Main GUI Components

8.2.1 Website

The website is only accessible to the Administrators of the website who also play the role of Rescue Coordinators.

Login

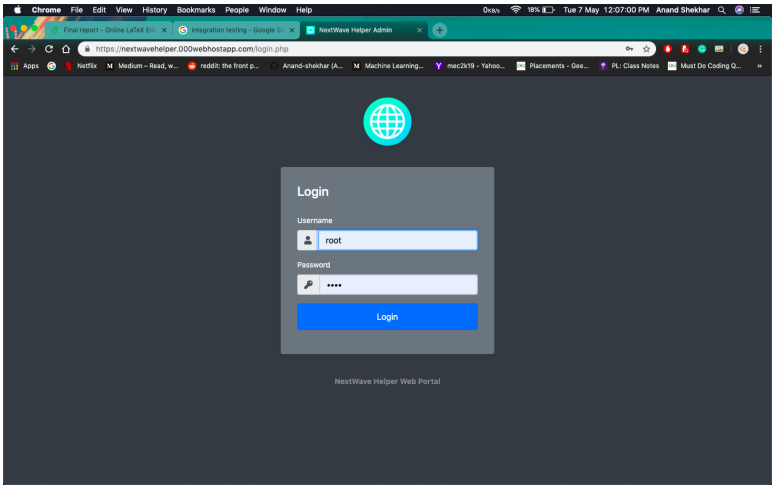

Figure 8.1: Login page

All administrators are required to verify their credentials by logging in.

Homepage

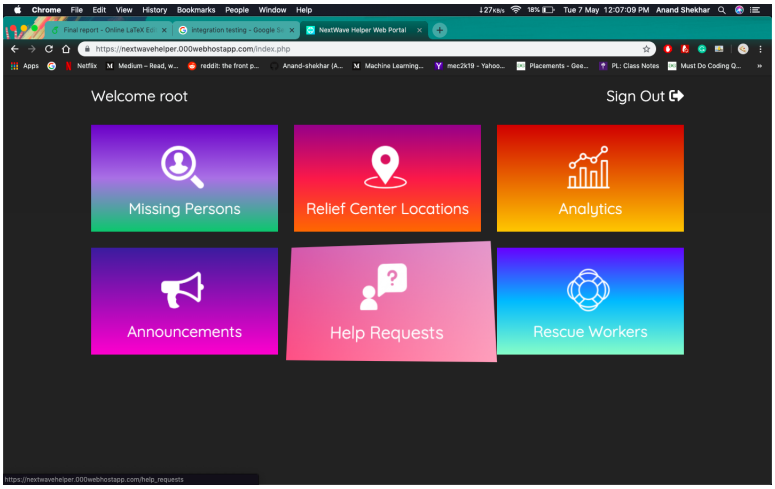

Figure 8.2: Home page

Once logged in, the admin is taken to the homepage where an array of features are at their disposal. This includes Adding announcements for the affected users, Managing Help requests and missing person, adding and reviewing relief centre locations, Analytics based on the rescue operation statistics and data and obtaining the list of registered rescue workers along with their location. Snaps of some those features are included below.

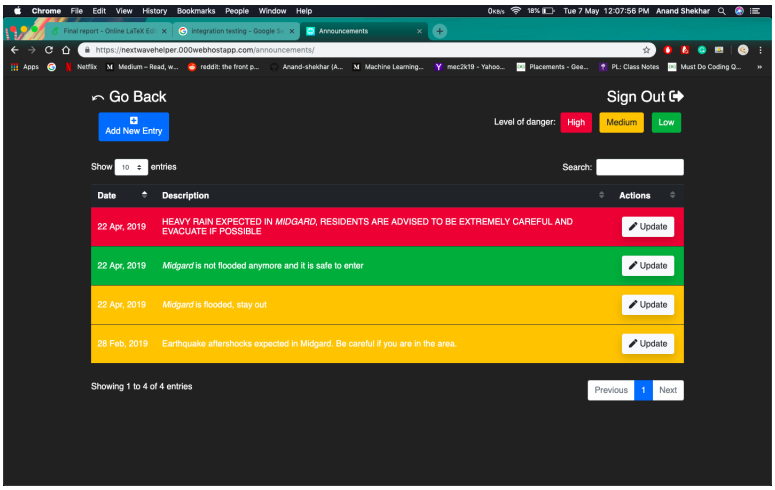

Figure 8.3: Announcements

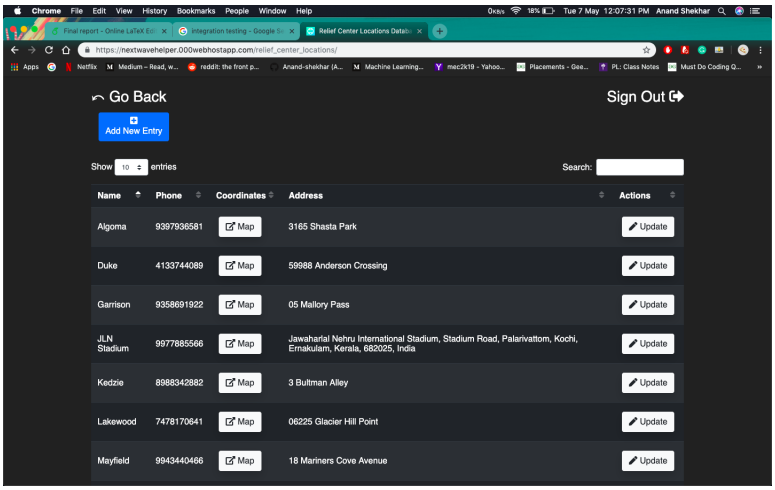

Figure 8.4: Relief Centre

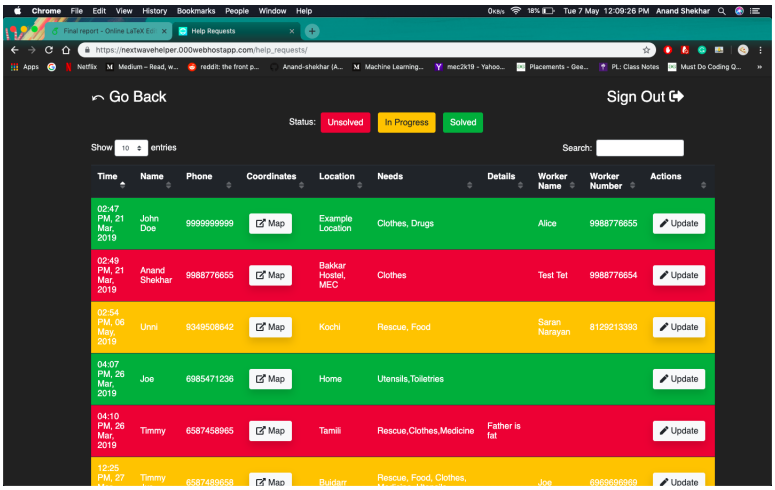

Figure 8.5: Help requests

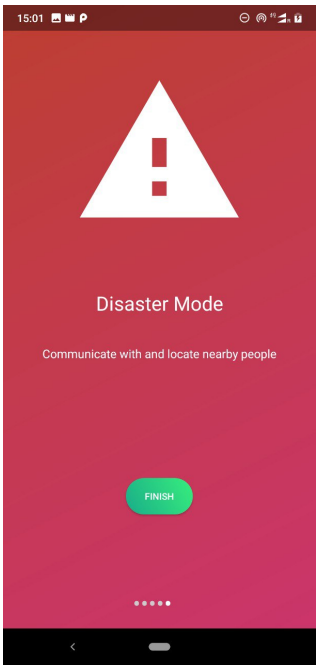

Figure 8.6: Intro slides

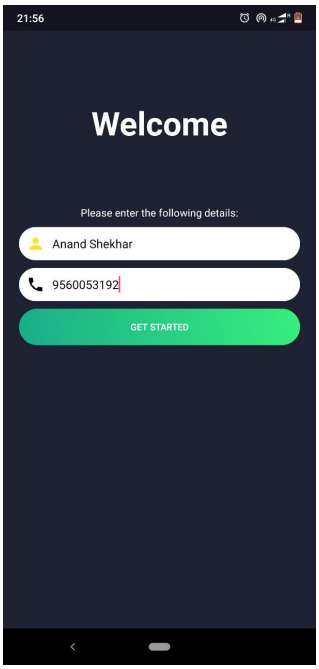

Figure 8.7: Welcome screen

8.2.2 Android Application

The core component of this project is the android application. On selecting the application, all new users are provided with a walkthrough of all the major features of the application. This is followed by a one-time login process.

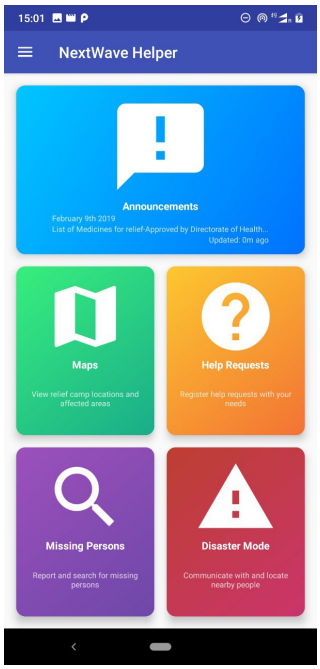

Figure 8.8: Dashboard

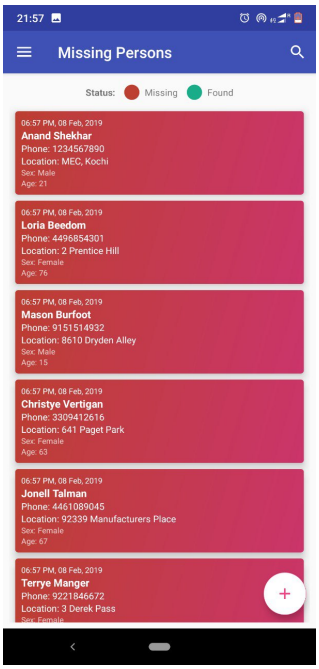

Figure 8.9: Missing person reports

Once logged in, the user is taken to the welcome page which has cards for all the available features on the app. The users have a plethora of options to choose from in this mode, which we have termed as the normal mode. In a disaster scenario, where communication lines are down, we have what is termed as a disaster mode. This mode only supports few features such as chat and location sharing.

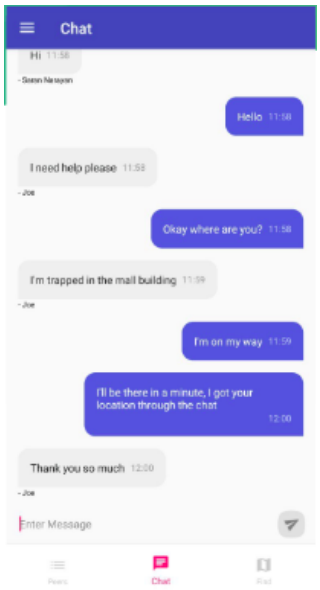

Figure 8.10: Group chat

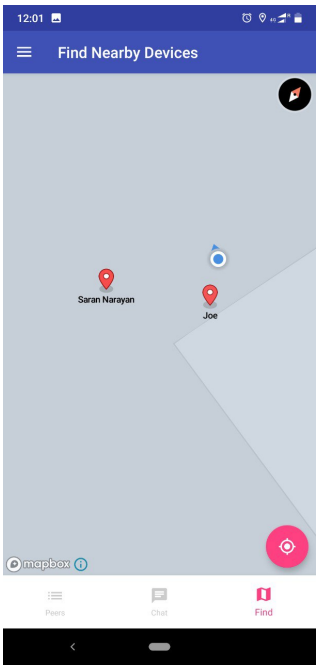

Figure 8.11: Location sharing

Here is a snippet of the chat feature along with location sharing.

Chapter 9

Results

For this demonstration, we'll consider one of the use case scenarios, which is requesting for help. The user starts off by adding a new help request.

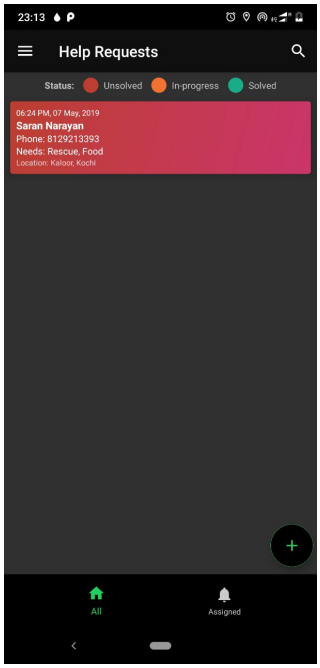

Figure 9.1: Help request section

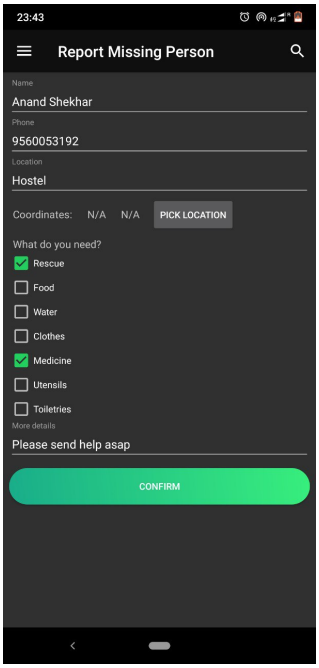

Figure 9.2: Request for help

Once a request is made, it is stored uniquely in the database. This request is fetched and displayed in help request section.

The new request is also available on the website, which is assigned to a rescue worker by the administrator based on their location.

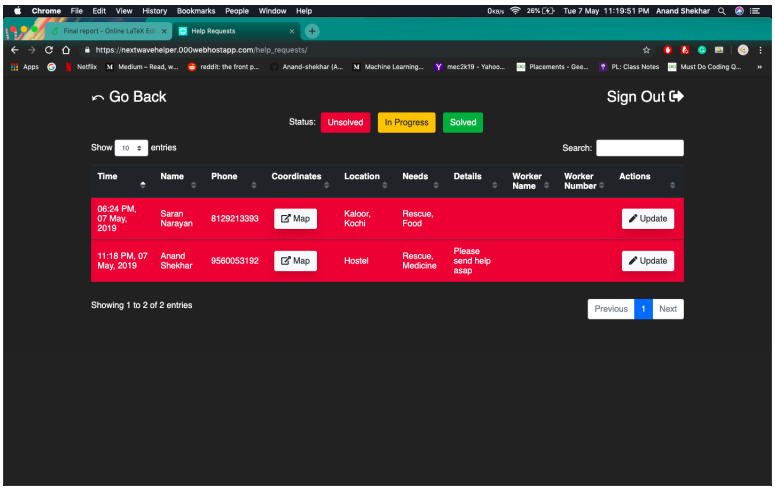

Figure 9.3: New help request

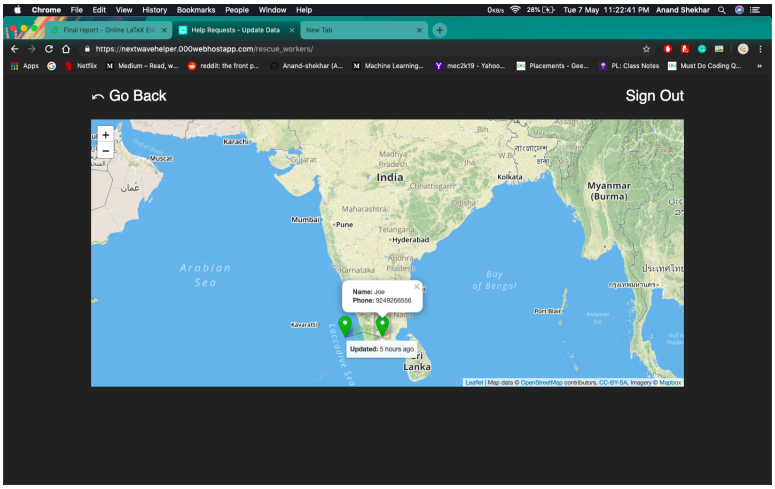

Figure 9.4: Rescue worker

The request is then assigned to an appropriate worker. Upon doing this, the request status changes to in-progress in both the website and application.

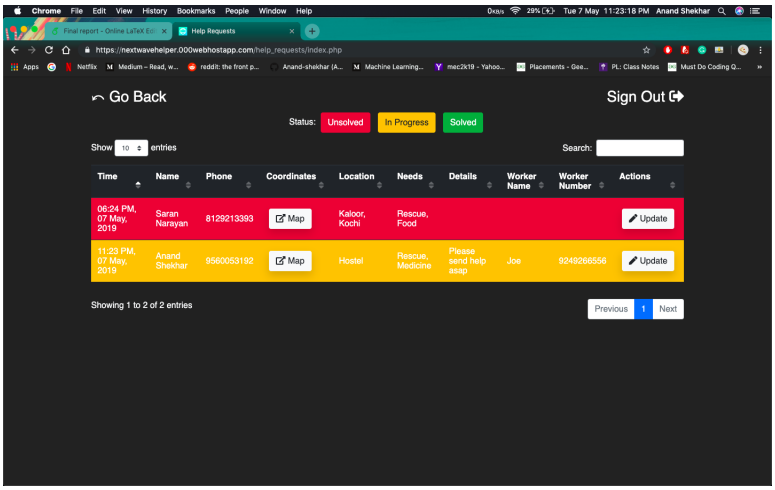

Figure 9.5: Help request assigned by rescue coordinator

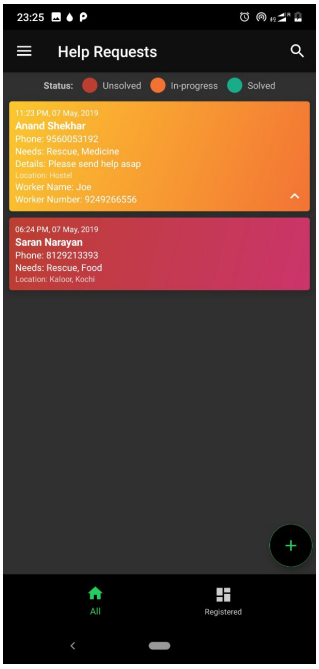

Figure 9.6: Help request assigned

The request in consideration was successfully stored, fetched and assigned to a worker for further process.

Chapter 10

Conclusion

We developed a fully functional application which can be used after a disaster strikes in order to aid communication and facilitate disaster recovery efforts. This can greatly improve communication between victims of the disaster and the people working to help them, leading to much more efficient use of time and resources. The project consists of 3 major parts:

1. The website, which is used by rescue coordinators to post new announcements, relief centres, update information about missing people, and assign help requests to rescue workers. These help requests are assigned after checking which rescue workers are closest to the location of the help request.

2. The application, which has 2 modes, normal mode and disaster mode. In normal mode, the user can:

    - View announcements
    - View maps, which contain information about relief camps and areas affected by the disaster
    - View help requests and request for help if they need to do so.
    - View missing people and report missing persons.

3. Disaster mode is meant to be used when normal modes of communication(such as phone and network signal) are unavailable. It allows users to chat and share locations with nearby users.

Users can also register as rescue workers within the app. This broadcasts their location every 5 minutes and allows the rescue coordinator to assign help requests to them. A user that has registered as a rescue worker is added to a list of rescue workers and can be seen on the website, from the administrator's side.

Chapter 11

Future Scope

The application can be extended to perform the following tasks.

1. Instead of using android phones to establish the offline communication network, it might be possible to use other, tougher devices which can withstand harsher conditions than your average smartphone to create the network, and have mobile phones connect to the network established by these devices. This can save power on the mobile phones, which could be used for other potential purposes.

2. In our current implementation of Help Requests feature, the help requests are manually assigned to the rescue workers by rescue coordinators using the web portal. Instead of manual assignment, the help requests could be automatically assigned considering the locations of the available rescue workers. This would increase the efficiency of the system.

Chapter 12

Publication

The International Journal of Innovative Research in Computer and Communication Engineering is a High Impact Factor, Open Access, International, Peer-Reviewed, Monthly journal that publishes original research articles in the fields of Computer Science, Information Technology and Communication Engineering.

The core vision of IJIRCCE is to propagate the innovative information and technology to promote the academic and research professionals in the field of Computer Science, Information Technology and Communication Engineering.

The paper was published in April 2019 edition of IJIRCCE.

Online : http://www.ijircce.com/upload/2019/april/41_NextWave.pdf

Authors : Manilal D.L, Anand Shekhar, Joe Davis, Saran Narayan
Pages : 8

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 4, April 2019

NextWave Helper: Android App for Search and Rescue

Manilal D. L.[1], Anand Shekhar [2], Joe Davis Akkara[3], Saran Narayan [4]

Head of Department, Department of Computer Science, Model Engineering College, Kochi, Kerala, India[1]

U.G Student, Department of Computer Science, Model Engineering College, Kochi, Kerala, India [2, 3, 4]

ABSTRACT: Natural and man-made disasters can have devastating effects on human society, but they can largely be mitigated with the coordinated efforts of rescue workers. However, existing disaster response means have multiple obstacles like accessibility, ease of use, dependency on social media and requirement of special skill sets on the part of the public. There can also be cases when the whole network infrastructure fails, which can seriously hinder rescue efforts and endanger the lives of the affected. To overcome these limitations, we developed an Android-based application that offers user-friendliness and real-time data related to crowd-sourced information, which can provide missing persons data, show affected areas and relief centre locations, and handle requests. In cases when the networks fail, we use Mobile Ad-hoc Networks (MANET) that will serve to establish and provide communication and coordination among mobile devices during the emergency situation.

KEYWORDS: MANET; emergency; disaster; Android; search; rescue

I. INTRODUCTION

In mid-August 2018, Kerala, a state in India was hit by one of the worst floods in its recent memory. With over 483 dead and more than a million displaced, it remains a dark specter that looms over the state even today. 3,200 relief camps were set up during and after the floods. The property damage is estimated to be around ₹40,000 crores (or US $5.6 billion). Soon after the tragedy struck, technical experts and volunteers came together and built the website 'keralarescue.in'[1] to obtain a platform to connect online calls for help with people coordinating offline rescue operations. The website[2] played an integral role in search and rescue operations and helped saved lives[3].

The website had a plethora of features, which we took inspiration from, like showing the latest announcements, help requests registered by people affected by the flood, locations of relief camps, and a list of missing and found people. However, being built in a hurry, the website was not as well designed as it could have been. The user interface was cluttered and not mobile ready, this affected the usability and made it difficult for senior citizens, who were primarily using their mobile phones to navigate the website.

During the floods, the communication in disaster-affected areas was erratic, with some damage and destruction of the communication infrastructure. In order to remedy this, we developed an android application for smartphones, which are carried by most people. It will have 2 modes of operation - a normal mode and a disaster mode. The normal mode will allow communication through normal channels, while the disaster mode will rely on an ad-hoc peer to peer network in order to allow communication even when infrastructure is damaged. This will allow the end-user to choose how to operate the application, based on the current situation. The app will also crowdsource information to mark areas affected by the disaster, which is an effective method to obtain information during trying times such as these.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 4, April 2019

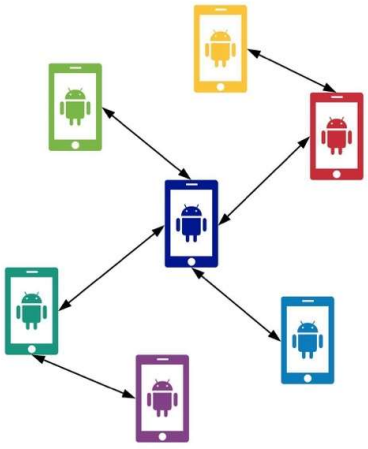

Fig 1. MANET formed by 7 devices

A mobile ad-hoc network (MANET) is an infrastructure-less self-configuring network, where each node acts as a router. There are no base stations, as in a normal network, and each node is free to move independently, making the network self-configuring and allowing devices in a MANET to leave or join the network dynamically. MANETs typically communicate at radio frequencies. All these features make a MANET a good solution to fix the communication issues that arise during disasters.

The rest of the paper is organized as follows. Section II discusses related works. Section III discusses the online features that are available in our application and the web portal, which allows people to coordinate with rescue workers to deliver supplies or find missing people. Section IV discusses the disaster mode feature, which creates a MANET with other devices in the area and its implementation. In Section V, we explain the algorithm used in this implementation. Section VI is concerned with the Performance review of the system. Finally, conclusions and plans for the future are discussed in section VII.

II. RELATED WORKS

A major issue is the possibility that communication infrastructure can be damaged during/after a disaster, leading to a loss of cellular connectivity or other standard means of communication. Several studies have been aimed to find remedies to this, but a large portion of them involve the deployment of additional infrastructure after the disaster, such as Satellite Access Points[4] and a wireless mesh infrastructure[5][6]. However, in case of a sudden, unexpected disaster such as an earthquake, it is often unrealistic to expect immediate mobilisation and deployment of this infrastructure and this delay, however slight, can lead to the loss of lives. Thus, it is imperative that an alternative to this which can be mobilised with almost no delay is developed.

[7] and [8] show that android applications are quite effective in aiding search-and-rescue efforts after a disaster, due to the number of people who have smartphones in this day and age. They show that usage of these applications can reduce response times of rescue workers and reduce obstructions in finding the victims, leading to lives saved. [9] and [10] show the value that crowdsourcing data can bring to situations such as these, with the lack of accurate information in the immediate aftermath of a disaster.

[11], [12] and [13] show the possibility of using a MANET in a situation where normal cellular connectivity is hampered and demonstrate its effectiveness. Another important feature discussed by these is the importance of reducing the power consumed by the application. Thus, implementing a MANET for mobile devices can be the solution to the possible inadequacy of communication methods after a disaster strikes. Along with this, implementing functionality

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 4, April 2019

that can be accessed via the internet can also help both victims and rescue workers with coordination and communication.

III. ONLINE FEATURES

With the current level of development of the internet, comprehensive internet connectivity is leading to an increasingly mobile reality. We are not tied to any single specific device, and everything is in the cloud. Further, coupled with Geographic Information System (GIS), extensive usage of these features has been made possible. This paper has two major components - the online web portal and android application. The web portal is only accessible to administrators and contains the backend of the application, while the Android Application is used by end users. There are 3 types of users throughout the paper the application:

1) Rescue Coordinator
2) Rescue Worker
3) Disaster Victim

Rescue coordinators are the administrators of the web portal who will receive information from the other 2 types of users through the internet and help to coordinate rescue efforts. Rescue workers will receive information from the coordinator through the application and work on their assigned help requests. Disaster victims can use the application to report/search for missing people, to register help requests, view map related information (relief camps locations, and areas affected by the disaster) and view relevant announcements.

A. *Android Application*:

The Android Application consists of 4 major parts, and is used by 2 types of end users - rescue workers and disaster victims.

1) Announcements: Announcements are posted by the rescue coordinator and users of the android application fetch these announcements through an Application Programming Interface (API). These announcements are cached in the application and updated whenever the app is opened and an internet connection is available (Fig. 2 Middle). The most recent announcement is also displayed in the home screen of the application.
2) Help Requests: Disaster victims can create requests for help, with their location (selected through a place picker on a map), and details of their requirements. The rescue coordinator can assign these requests to a rescue worker, who will try to work on them. The rescue worker can see the list of all registered help requests, and also help requests which have been assigned to him. 'Solved', 'Unsolved' and 'In-progress' help requests are assigned different colours (green, red and yellow respectively) when displayed in the application.
3) Missing Persons: Missing people can be reported and updated in the database. Once the person has been found, their status is updated by the rescue coordinator. 'Missing' and 'Found' people are assigned colours (red and green respectively) when displayed in the application (Fig. 2 Left).
4) Information Maps: The application also features maps which show a variety of information such as:
    - Affected Locations: Locations that are affected by the disaster can be marked on the map. In order to ensure that people don't maliciously mark locations as unaffected (Fig. 2 Right), we've implemented this in a way such that only people close to the area can mark it as affected. On selecting a road to mark as affected, the database will be updated and everyone who views the map will be able to see that particular road as an affected area.
    - Relief Camps: Relief camp locations can also be viewed in the application. The rescue coordinator must add relief camps through the web portal, following which they will be visible in the application. The user can view a list of relief camps, in which the closest camps will be displayed first. The user can also click on items in the list to view them on the map, or just view a map of their location, with nearby relief camps marked on it.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 4, April 2019

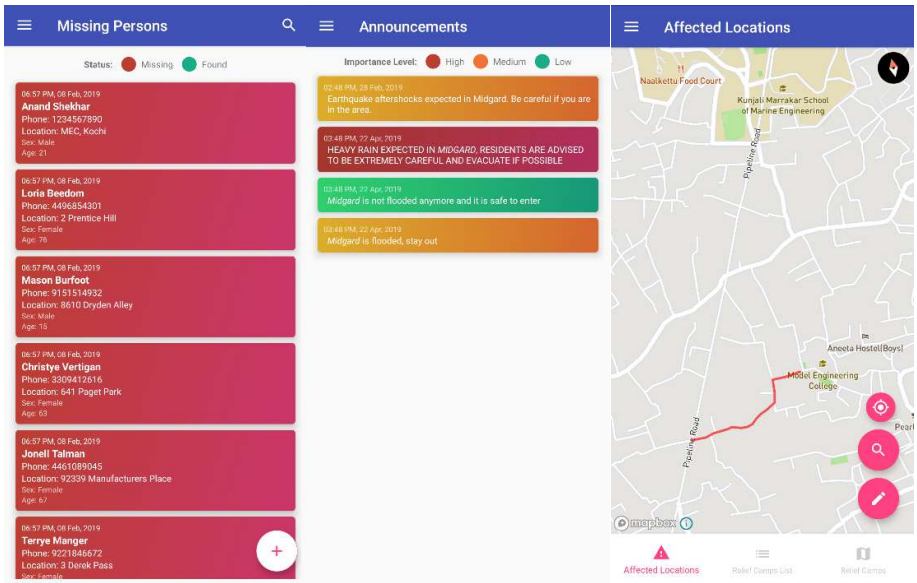

Fig.2. The android application

B. *Online Admin Portal*

The admin portal has an API used to upload data from end users and an interface for rescue coordinators to interact with the help requests - to change statuses of completed requests, missing persons or assign help requests to rescue coordinators. Other than adding announcements and relief camps, and marking missing people as found, the most important job for the rescue coordinator is to manage help requests.

1) Managing Help Requests: The rescue coordinator can view the help request on Google Maps, and then assign it to a particular rescue worker using the rescue worker's phone number (Fig. 3). The rescue worker can then view those help requests in the mobile application under the 'Assigned' section in "Help Requests". When allocating the help request to a rescue worker, the rescue coordinator sets the status of the request from 'Unsolved' to 'In-progress', letting the user know that their request has been assigned to a rescue worker.

2) Analytics: For visualization past data and increased accessibility, there is an option for representing the data from the website in the form of doughnut charts. In the analytics section of the website, the admin can view the history of requests - either all the requests together, or a subset of requests within 2 dates. The history of the selected request is shown, along with a doughnut chart to show the percentage of requests of different types - Solved/In-Progress/Completed in the case of help requests, or Missing/Found in the case of missing people. This allows us to analyse what fraction of the requests remained unfulfilled, and try and work out areas where requests remain uncompleted, for whatever reason and take steps to improve on them.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 4, April 2019

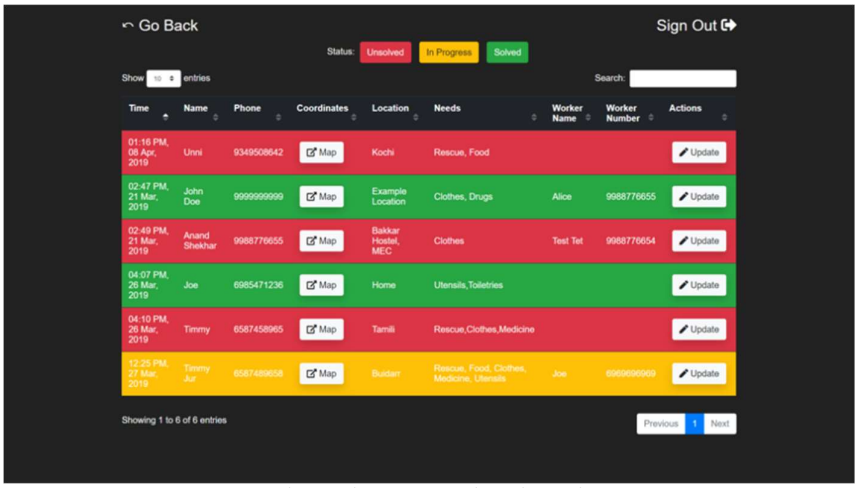

Fig.3. Help requests on the web portal

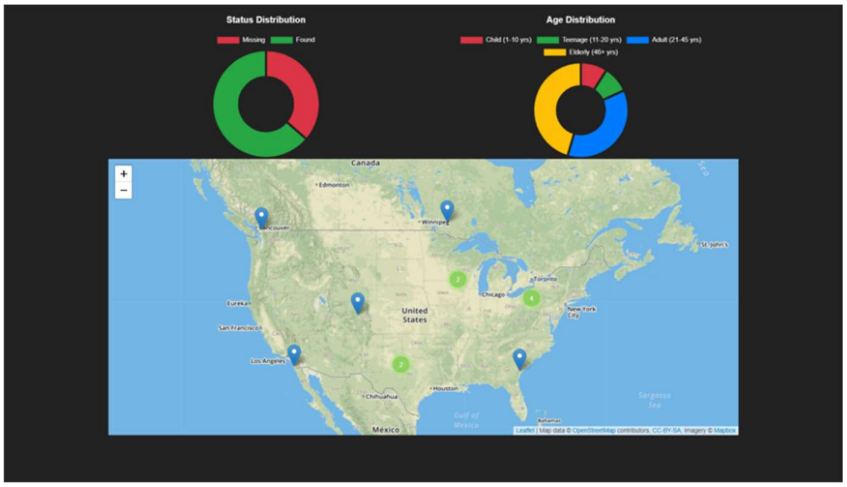

Fig.4.Analytics for missing persons on web portal

It is also possible to view the requests in cluster form on the map (Fig. 4), to identify where exactly the requests are coming from - to identify which areas have a high volume of requests and focus more efforts on those areas.

IV. OFFLINE MODE

The offline functionality of Application mainly revolves around Disaster mode. This mode employs the Google Nearby Communications API, which uses Bluetooth and Wi-Fi to search for and locate nearby peer devices. All the features are made available, once connection is established between 2 or more peer devices.

A. *Mobile Ad-hoc Network (MANET)*

A MANET is a type of ad-hoc network that can change locations and configure itself on the fly, and is a Delay Tolerant Network (DTN). Because MANETS are mobile, they use wireless connections to connect to various networks. They consist of set of mobile nodes connected wirelessly in a self-configured, self-healing network without having a fixed infrastructure. MANET nodes are free to move randomly as the network topology changes frequently. Each node

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 4, April 2019

behaves as a router as they forward traffic to other specified node in the network. This can be used in road safety, sensors for environment, home, health, disaster rescue operations, air/land/navy defense, weapons, robots, etc. We are using this network type for offline communication.

B. *Communication using MANET*

The steps to begin communication are as follows, the final screenshot is show in Fig. 5:

1) Switch to the Disaster Mode section of the application.
2) Click on the Discover Peers button in order to discover other devices that are connected/available to connect to the MANET.
3) Once the peers are discovered, the window will automatically switch to the chat.
4) Now, the user can send messages to other users connected via the MANET.
5) The user can also view the locations of other people in the chat by switching to the 'Find' tab.

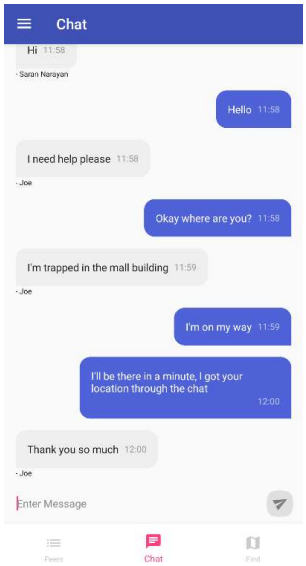

Fig.5.Disaster mode for offline chat

C. *Locating People using MANET*

There is also a provision to locate the people currently connected to the MANET, which can be accessed simply by switching to the 'Find' tab after connecting to a peer. This opens a map, which shows the location of the connected members of the MANET.

D. *Description of the Proposed Algorithm:*
The following algorithm highlights the basic steps involved in setting up the P2P connection between two Android devices using the Nearby Connections API.

Step 1: Start
Step 2: startAdvertising() //starts advertising service to nearby devices
Step 3: startDiscovering() //starts discovering for nearby devices
Step 4: if onEndpointFound() called then

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 4, April 2019

```
Step 5:  stopDiscovering()                    //stops discovering nearby devices
Step 6:  requestConnection()                  //requests connection to the found endpoint
Step 7:  if acceptConnection() then
Step 8:          onConnectionResult() callback   //This is called on both devices
Step 9:          Connection established
Step 10: else if rejectConnection() then
Step 11:         startDiscovering()           //starts discovering for any other devices
Step 12: End
```

V. PERFORMANCE REVIEW

A. Discovery time

In the discovery phase, time elapsed between initialization and the point at which all nodes are discovered is measured. We found in our results (Fig. 6) that delay increases non-linearly. This test was done with less than 10 peer devices. Any number of peers greater than that wasn't in the scope of our project. This behavior was observed due to the non-discovery of nearby devices in the first.

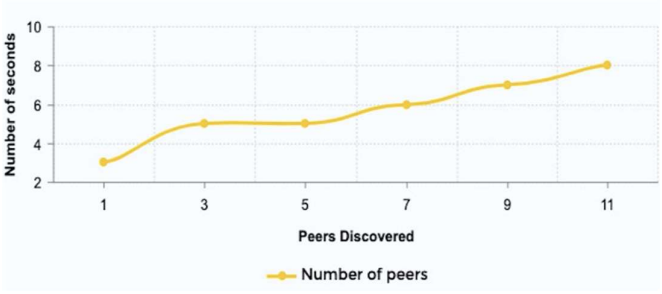

Fig. 6. Time to discover peers vs Number of peers

B. Bandwidth comparison

We tested the bandwidth of offline communications with two devices (Moto G5 Plus - Bluetooth 4.2 and Redmi 3S - Bluetooth 4.1). One device would act as the master and the other as slave. The slave echoes the messages it receives and was kept fixed at one location. This scenario gave us the round-trip time (RTT) from which we can calculate the speed in bytes per second easily.

While testing we noticed that there is little effect on the bandwidth for distances up to 100 feet. Also, that if we transmitted a 5KB message (two paragraph messages, which is unrealistic in our situation) the time remained almost constant. So, we can conclude that it has high bandwidth and the time it takes to transmit is more or less the delay to establish a communication link.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 7, Issue 4, April 2019

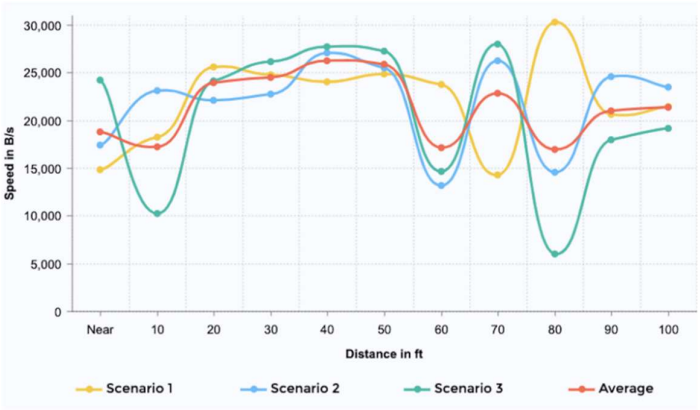

Fig. 7. Bandwidth comparison for different distances for a 1024 byte message

The three scenarios mentioned in the graph (Fig. 7) is just three messages sent at the same distance but at different time intervals to get an average because of the variations we observed in the values for the same distances.

VI. CONCLUSION

In this paper, we developed an Android based application for smartphones, which can be used after a disaster to help the search and rescue efforts. We demonstrate its features, namely the ability to view announcements, add missing people and request for help and view affected areas and relief camps on the map. The application also allows for offline communication during disaster, which is for a situation where the standard communication infrastructure is damaged and the disaster victim cannot receive a mobile phone signal.

REFERENCES

1. IEEEKeralaSection, "Source code forhttps://keralarescue.in",Retrieved fromhttps://github.com/IEEEKeralaSection/rescuekerala.
2. Govt. of Kerala, Kerala State IT Mission and IEEE Kerala Section, "Kerala Rescue", Retrieved from https://keralarescue.in.
3. Shahim Baker, "Volunteers Come Together to Help Survivors of Floods in Kerala, India", Retrieved from http://sites.ieee.org/sb-uol/volunteers-come-together-to-help-survivors-of-floods-in-kerala-india/
4. F. Patricelli, J. E. Beakley, A. Carnevale, M. Tarabochia, and D. K.Von Lubitz, "Disaster management and mitigation: the telecommunications infrastructure", Disasters, vol. 33, no. 1, pp. 23–37, 2009.
5. R. B. Dilmaghani and R. R. Rao, "A wireless mesh infrastructure deployment with application for emergency scenarios", in 5th InternationalISCRAM Conference,Citeseer, 2008.
6. S. Mathur, "A rapidly deployable communications network architecturefor disaster management",Citeseer, Tech. Rep., 2009.
7. A. Hossain, S. K. Ray and R. Sinha, "A Smartphone-Assisted Post-Disaster Victim Localization Method", 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Sydney, NSW, 2016, pp. 1173-1179, 2016.
8. K. M. Rahman, T. Alam and M. Chowdhury, "Location based early disaster warning and evacuation system on mobile phones using OpenStreetMap", 2012 IEEE Conference on Open Systems, Kuala Lumpur, pp. 1-6, 2012.
9. Jane C. Samonte, Mary M. Rozario, Raymond Cleo, B. Aranjuez Nieland A. Maling Christopher,"Crowdsourced mobile app for flood risk management", pp 65-71, 2017.
10. L. Ma, X. Chen, Y. Xu, Y. Gao and W. Liu, "Study on crowdsourcing-compatible disaster information management system based on GIS", 2014 International Conference on Information Science, Electronics and Electrical Engineering, Sapporo, pp. 1976-1979, 2014.
11. HossmannTheus,Schatzmann Dominik, Carta Paolo and Legendre Franck,"Twitter in disaster mode: smart probing for opportunistic peers", Proceedings of the Third ACM International Workshop on Mobile Opportunistic Networks - MobiOpp '12, pp 93-94, 2012.
12. H. Yuze, Y. Qian and N. Suzuki, "Development of Smartphone Application for Off-Line Use in Case of Disaster", 27th International Conference on Advanced Information Networking and Applications Workshops, Barcelona,pp. 243-248, 2013.
13. S. Bhattacharjee, S. Kanta, S. Modi, M. Paul and S. DasBit, "Disaster messenger: An android based infrastructure less application for post disaster information exchange"2016 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), Bangalore, pp. 1-5, 2016.

References

[1] V. Tundjungsari and A. Sabiq, "Android-based application using mobile adhoc network for search and rescue operation during disaster," 2017 International Conference on Electrical Engineering and Computer Science (ICECOS), Palembang, 2017, pp. 16-21.

[2] https://keralarescue.in

[3] https://github.com/IEEEKeralaSection/rescuekerala

[4] http://sites.ieee.org/sb-uol/volunteers-come-together-to-help-survivors-of-floods-in-kerala-india/

[5] A. Hossain, S. K. Ray and R. Sinha, "A Smartphone-Assisted Post-Disaster Victim Localization Method," 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Sydney, NSW, 2016, pp. 1173-1179. doi: 10.1109/HPCC-SmartCity-DSS.2016.0164

[6] H. Yuze, Y. Qian and N. Suzuki, "Development of Smartphone Application for Off-Line Use in Case of Disaster," 2013 27th International Conference on Advanced Information Networking and Applications Workshops, Barcelona, 2013, pp. 243-248.

[7] L. Ma, X. Chen, Y. Xu, Y. Gao and W. Liu, "Study on crowdsourcing-compatible disaster information management system based on GIS," 2014 International Conference on Information Science, Electronics and Electrical Engineering, Sapporo, 2014, pp. 1976-1979.

[8] Jane C. Samonte, Mary & M. Rozario, Raymond & Cleo B. Aranjuez, Niel & A. Maling, Christopher. (2017). Crowdsourced mobile app for flood risk management. 65-71. 10.1145/3162957.3162962.

[9] Hossmann, Theus & Schatzmann, Dominik & Carta, Paolo & Legendre, Franck. (2012). Twitter in disaster mode: smart probing for opportunistic peers. 10.1145/2159576.2159601.

[10] K. M. Rahman, T. Alam and M. Chowdhury, "Location based early disaster warning and evacuation system on mobile phones using OpenStreetMap," 2012 IEEE Conference on Open Systems, Kuala Lumpur, 2012, pp. 1-6. doi: 10.1109/ICOS.2012.6417627

[11] S. Bhattacharjee, S. Kanta, S. Modi, M. Paul and S. DasBit, "Disaster messenger: An android based infrastructure less application for post disaster information exchange," 2016 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), Bangalore, 2016, pp. 1-5.