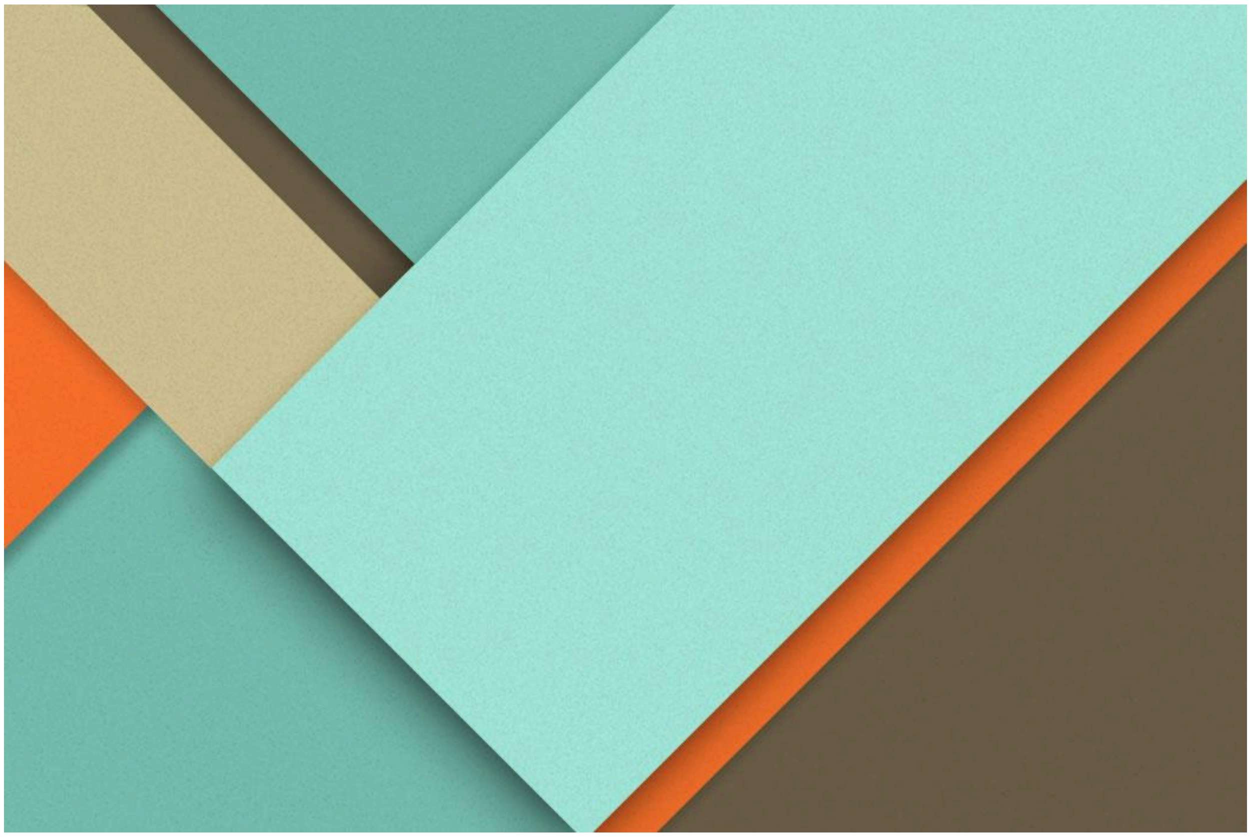


Software Requirements Specification

for

Disaster Preparedness Mobile App

Version 1.0

Prepared by

Group Number: 7

Rudr Prasad	S11219309	100%
Shamal Prasad	S11219545	100%
Nikhil Kumar	S11195943	100%
Pranav Kumar	S11219649	100%

Course: CS218

Date: *03/10/2024*

Table of Content

1. Introduction **4**
1.1 Purpose 4
1.2 Document Conventions 4
1.3 Intended Audience and Reading Suggestions 4
1.3.1 Stakeholders 4
1.3.2 System Developers 5
1.3.3 Quality Assurance Engineers 5
1.3.4 Maintenance Engineers 5
1.4 Product Scope 5
1.5 References 6
2. Overall Description **6**
2.1 Product Perspective 6
2.2 Product Functions 9
2.3 User Classes and Characteristics 9
2.3.1 User Class 9
2.3.2 Likely User Class 10
2.4 Operating Environment 10
2.5 Design and Implementation Constraints 11
2.5.1 Software Limitations 11
2.5.2 Hardware Limitations 11
2.6 User Documentation 11
2.7 Assumptions and Dependencies 12
3. External Interface Requirements **13**
3.1 User Interfaces 13
3.1.1 Splash Screen 13
3.1.2 Registration Screen 14
3.1.3 Login Screen 15
3.1.4 Home Screen 16
3.1.5 ToDo List and Disaster Tips Screen 17
3.1.6 Relief Shelter Location 18
3.1.7 Emergency Contact Screen 19
3.1.8 Live Disaster Tracking 20
3.1.9 Notification tray 21
3.1.10 Forgot Password 22
3.1.11 Change Password 23
3.1.12 Admin Menu Screen 24

3.1.13 Create Disaster Tips (Admin Account) 25
3.1.14 Add Emergency Contact (Admin Account) 26
3.1.15 Create Disaster Alert (Admin Account) 27
3.1.16 Create Relief Shelter (Admin Account) 28
3.2 Hardware Interfaces 28
3.3 Software Interfaces 29
3.4 Communications Interfaces 29
4. System Features 30
4.1 Creation, Update and Deletion of Information Regarding Disaster Action 30
4.1.1 Description and Priority 30
4.1.2 Stimulus/Response Sequences 30
4.1.3 Functional Requirements 31
4.2 Creation, Update and Deletion of Relief Shelter Location 31
4.1.1 Description and Priority 31
4.1.2 Stimulus/Response Sequences 32
4.1.3 Functional Requirements 32
4.2 Push Disaster Alerts to mobile devices 32
4.1.1 Description and Priority 32
4.1.2 Stimulus/Response Sequences 33
4.1.3 Functional Requirements 33
4.2 Track Real Time Location of Disaster 33
4.1.1 Description and Priority 33
4.1.2 Stimulus/Response Sequences 33
4.1.3 Functional Requirements 34
5. Other Nonfunctional Requirements 34
5.1 Performance Requirements 34
5.1.1 Response Time 34
5.1.2 Efficiency 34
5.1.3 Resource Utilization 34
5.1.4 Storage Capacity 34
5.1.5 Battery Consumption 34
5.1.6 Error Recovery 34
5.2 Safety Requirements 35
5.3 Security Requirements 35
5.4 Software Quality Attributes 35
5.4.1 Adaptability 36
5.5 Business Rules 37
6. Other Requirements 37
Appendix A: Glossary 37

Appendix B: Analysis Models 38
Appendix C: References 39

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of this document is to provide a detailed description for developing the Disaster Preparedness Mobile App. The document outlines the project scope, motivation for the project, features or functional requirements along with its constraints, various user and system interfaces, architectural design, data management techniques and response to external factors.

With the following information, this document is meant to serve as a reference to stakeholders, developers, testers and system maintainers to ensure consistent understanding of the project's scope and structure.

1.2 Document Conventions

In the following document, headings, sub-headings, and normal text have been differentiated by its font size. Where a heading is of size 16 and bolded, subheading is size 14 and bolded, while normal text is size 11 unbolded. All main headings have been numbered as 1,2,3 and so forth and the following subheadings under its respective headings have been numbered as 1.1,1.2,1.3 and so forth; any further subheadings of subheadings or detailed subheadings will be labeled as 1.1.1,1.1.2,1.1.3 and so forth. Main Headings are not indented at all and so aren't any other normal content, however subheadings and following subheadings are indented.

1.3 Intended Audience and Reading Suggestions

The SRS document is prepared to cater to the needs of various users of the document before, during and after the development of the system.

1.3.1 Stakeholders

Stakeholders can use the SRS document as a source of contract that outlines all features and functionalities that the development team has agreed to implement into the final project. The information that the stakeholders may be interested in would be the User interface, the functional requirements and nonfunctional requirements, therefore they would be mostly interested in headings 3, 4 and 5 that outline this information.

1.3.2 System Developers

The system developers use the SRS document to read on what features that need to be implemented in the project and the constraints placed on the development and implementation of the final product. The information of interest for system developers would be functional requirements, nonfunctional requirements, interface requirements and UML diagrams to act as references during implementation of the project. The following information can be found under heading 3, 4,5 and 6.

1.3.3 Quality Assurance Engineers

The quality assurance engineers use the SRS document to check whether the developed product/project matches the features, functionality and constraints that was originally set during the planning of the project. QA engineers need the information on interface requirements, functional requirements, constraints and UML models to ensure consistency with the project proposal and implementation of the project. The necessary information can be found under headings 3,4,5 and 6.

1.3.4 Maintenance Engineers

The maintenance engineers use the SRS document to comprehend functionalities of the system and any constraints. Maintenance engineers would require functional requirements, nonfunctional requirements, interface requirements, as well as user documentation that would need to be adjusted after maintenance of the program. The interested information could be found under heading 3,4,5 and 2.6.

1.4 Product Scope

The Disaster Preparedness mobile system acts as a comprehensive disaster preparedness and alerting solution that allows the government's disaster management sector and meteorological department to send natural disaster safety tips before and after a natural disaster strikes and send real-time alerts to app users, informing them of the current status of the disaster.

The objective of such an app is to bolster the disaster preparedness and disaster response to maximize the safety of every citizen of Fiji through timely information that is complete and reliable.

To unravel the scope of the mobile system, the disaster preparedness app relies on a client/server architecture, where the admin user comprises members from the Fiji Meteorological service and the Department of Disaster Management Fiji. The admin team can create, update and delete disaster specific safety tips and information. The admin team can also create, update and delete relief shelter locations and its details. Having up- to-date information on the development of the natural disaster is paramount, therefore the admin team can provide real-time notification to inform users of the development of the disaster through push notification. The app will also display the approximate location of the disaster that users can track through a disaster map and be updated on the whereabouts of the disaster.

The end users will receive all information about the natural disasters through the app and act accordingly to ensure safety for themselves and their families. The disaster preparedness app provides multitudes of benefits to the end users. Users will have timely information that will assist them in making informed decisions and be better equipped to deal with natural disasters. Users will also be able to locate shelters in case of any disaster that places them in danger in their current location.

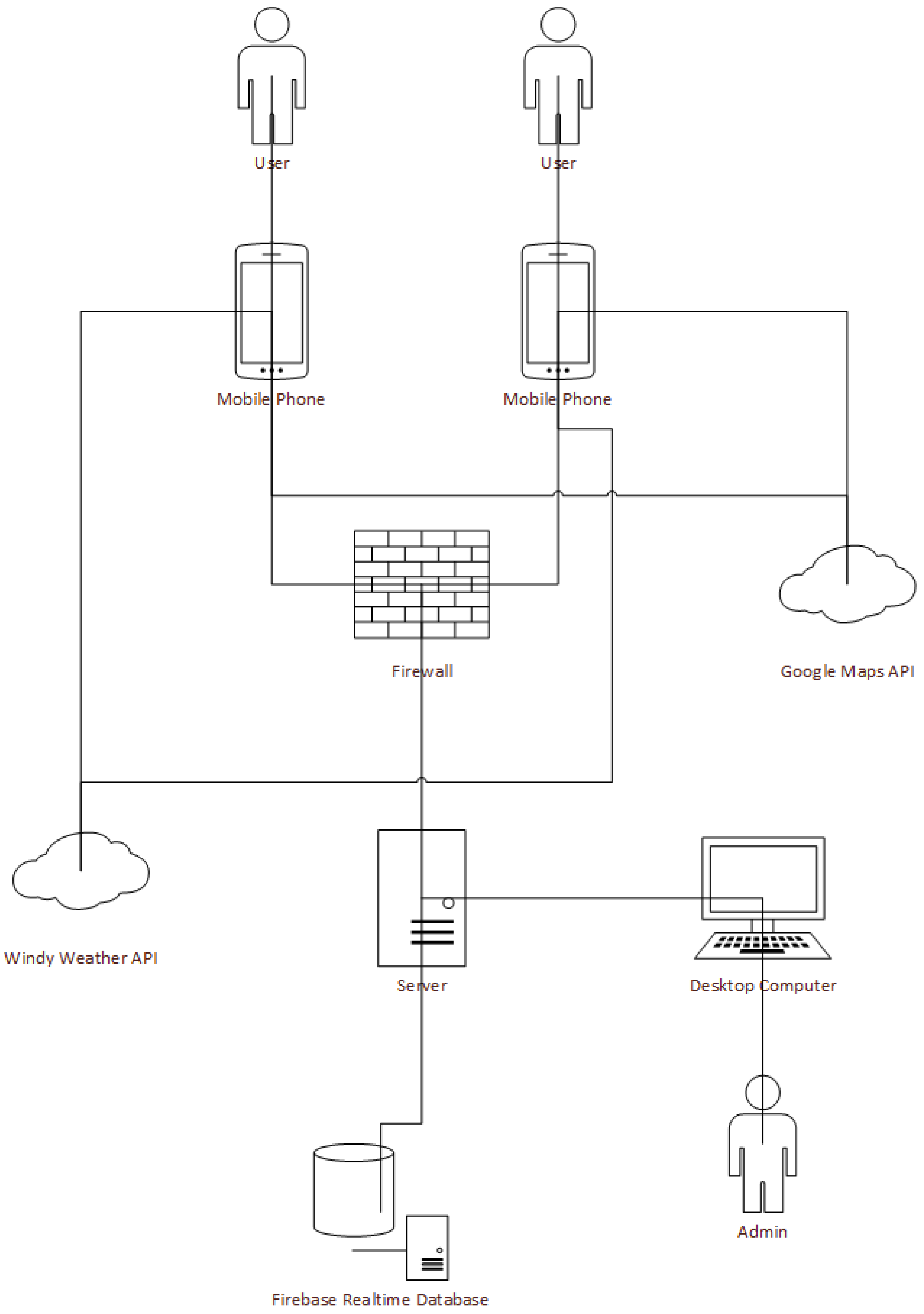
1.5 References

Prasad, S. (2024, September 25). *CS218-A2.vsdx* [System functionality and system architecture].

2. Overall Description

2.1 Product Perspective

The Disaster Preparedness mobile app is a first of its kind for Fiji, where the system that shall be developed is a new self contained product that bundles together disaster tracking, alerting and information distribution to all users of the system. The system borrows services from firebase real time database manager to send disaster alerts in real time. Moreover, the mobile system uses Google Maps API to show the location of shelters on Google Maps. Furthermore, the system also communicates to the Windy API to showcase real time location of disasters, mainly cyclones and weather patterns.


User
User
Mobile Phone
Mobile Phone
Firewall
Google Maps API
Windy Weather API
Server
Desktop Computer
Admin
Firebase Realtime Database

2.2 Product Functions

The following are the high level features and functionalities of the Disaster preparedness mobile app:

- Admin creates,updates and deletes various disaster safety and survival tips.
- Admin adds, updates or deletes relief shelters from the app.
- Users view various disaster preparedness information.
- User views the location of relief shelters on the google map.
- Admin pushes disaster alerts to the user's device.
- Users view the real time location of disaster and weather pattern in a weather map.

2.3 User Classes and Characteristics

The product is designed to be accessible and usable to a large demographic of users with different educational levels and various age groups with varying proficiency when it comes to using digital technologies.

2.3.1 User Class

The product is designed to be used by two user classes, namely the admin team, who are members from the Fiji meteorological Station and Department of Disaster Management, and the end user, which involves the entire general public.

The admin team consists of users with a high level of technical and analytical ability that help determine the best courses of actions in case a natural disaster occurs. This subset of users are highly educated, experienced and possess a high level of security privileges to create, update and delete information that deals with the safety and lives of the users who rely on this information.

The larger subset of users are the general public who have the most diverse levels of technical expertise, educational levels, living conditions (eg, location, shelter structure, and family dynamics) and experience with dealing with disasters before it occurs and how to react during or after the disaster occurs. Tech savvy users can make the most of the information and features the Disaster Preparedness app provides to ensure safety, while the lesser tech savvy users can use the app as an alerting system to notify themselves of the development of the disasters.

2.3.2 Likely User Class

The application is designed to be used by a small group of users identifying as the admin and the larger part of the applications user base is composed of end users that could simply download the mobile app from the play store.

The mobile app is designed with the larger subset of users in mind, which are the general public, as these are the users who need the information pertaining to the plan of action before, during and after a natural disaster.

The secondary set of users are the admins who will occasionally update information present in the app when the need arises.

In this manner, the class of users the Disaster preparedness app is developed to satisfy are the general public who need access to various information at all times.

2.4 Operating Environment

The minimum requirement of the Disaster Preparedness mobile app consist of the following:

- Ram: 2GB or higher
- Storage: 300MB
- OS: Android 8.0 (Oreo) or higher
- Network: 3G minimum, Wi-Fi recommended for large data transfers
- Sensors: GPS
- Device Type: Mobile/Handheld Devices

2.5 Design and Implementation Constraints

2.5.1 Software Limitations

The disaster preparedness mobile app is required to integrate a localized database that will store all information in cases of network disconnectivity. This for itself is not a limitation, however as per stakeholder requirement, the system is to integrate SQLite Database and no other variation of database technology.

The software is not designed to be completely operational offline and therefore will need to periodically connect to the server to access up to date information. Access to the external systems will also require internet connectivity.

2.5.2 Hardware Limitations

The Disaster Preparedness mobile app relies on a client/server architecture where the admin stores information on the server and the user mobile app needs to periodically connect to access information that is consistent and reliable.

The mobile app also uses 2 distinct external system interfaces to provide certain functionality to the program. The system uses the Google Maps API to show locations of relief shelters on the map and its description. In the exact manner, the system uses Windy API to show a disaster map that showcases weather pattern and real time disaster location.

The application needs a real time database and message controller that allows information from admin users to reach end users. To achieve this service, Firebase is used to allow real time database management on server side and message system to send information and alerts to end user mobile devices.

2.6 User Documentation

All the features and functionalities of the mobile app will be explained through the app itself. However, there will be a frame present that will act as a support frame containing information that would explain how to contact customer support. Either through email or a toll free number. Moreover, a user guide can also be provided through the support page that will explain to users all the features of the mobile app and how to correctly use these features in a reliable manner. The user guide will be a pdf file that can be opened by an external app that the Operating System deems suitable to open the file type.

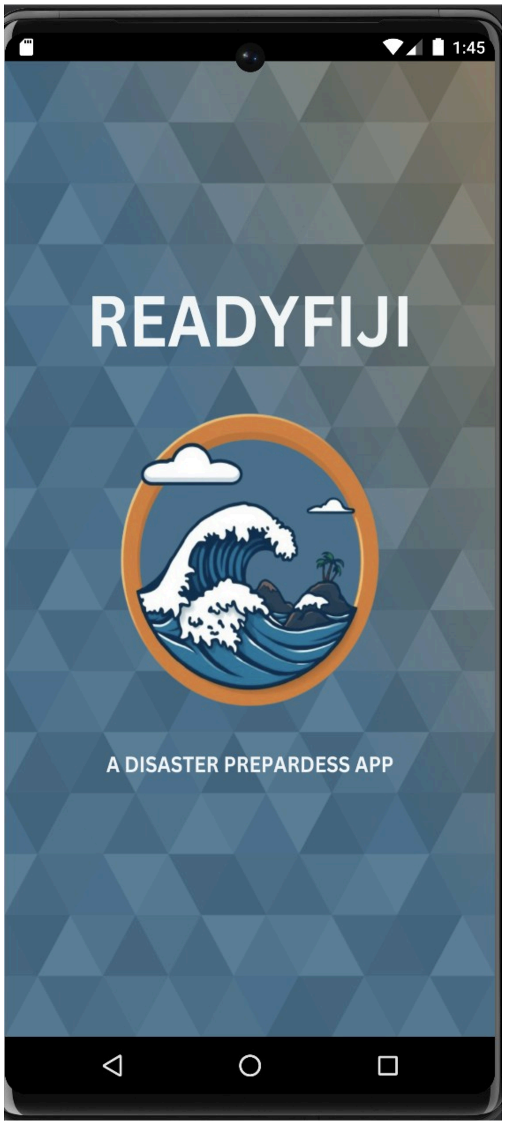
2.7 Assumptions and Dependencies

The development of the Disaster Preparedness mobile app relies on a number of external technologies to ensure the system is able to deliver some of its core features. The system plans to integrate Google Maps interface into the mobile app to show relief shelter locations. This service depends on whether the API allows the development team to correctly implement this particular feature as there can be concerns regarding the compatibility with the SDK version used to develop the mobile app. Moreover, the system relies on an external weather forecast providing service to provide the app users access to a disaster map that shows weather pattern and real time disaster tracking. This service is provided by Windy, a weather forecasting service that provides a weather map which includes cyclone tracking, demonstrates weather patterns and predicts future forecasts. This particular service is implemented through either using an API or embedding a website fragment into the mobile app. The actual implementation of this service through the following external system has yet to be determined as we do not know if API keys provided are free or whether the embedded website fragment fulfills the service that it is supposed to provide. Therefore, moving forward, the forecasting service interface may change to another external system that can provide the service as required. Moreover, the mobile app relies on a client/server architecture where a service is required to store common data from the admin user and update the information on the end user mobile app. The server should also be used to push alerts to the mobile phone when a natural disaster is formed. The implementation of the following service is yet to be determined, but some technologies that were considered were using Firebase as a server to handle real time database executions and asynchronous message exchange between admin user and end user. Other alternatives that were discovered were using an external Android Studio library called Retrofit to handle network requests between nodes. However, further research is required in the designing and implementation of this particular service.

3. External Interface Requirements

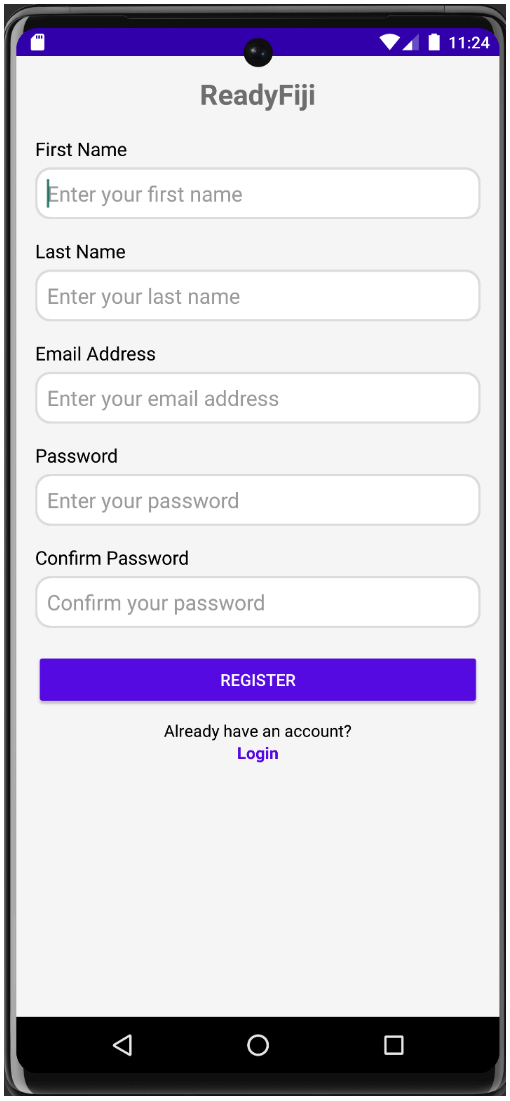
3.1 User Interfaces

3.1.1 Splash Screen



This is the mobile app's splash screen that will be displayed for a few seconds when the app is launched. This gives time for other background components to load and for the app to connect to the server through a network.

3.1.2 Registration Screen


11:24

ReadyFiji

First Name

Enter your first name

Last Name

Enter your last name

Email Address

Enter your email address

Password

Enter your password

Confirm Password

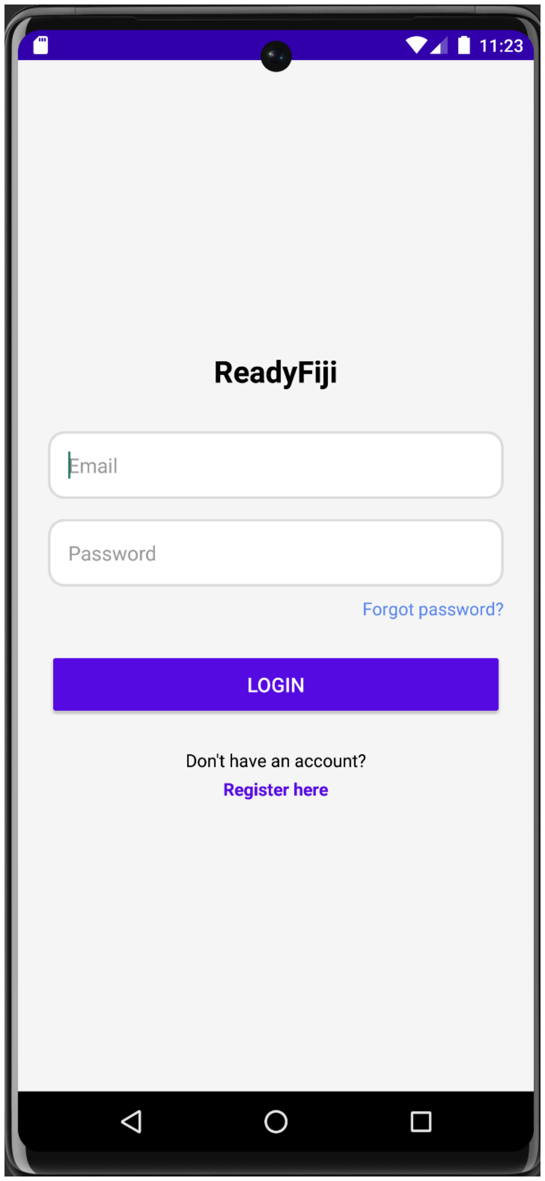
Confirm your password

REGISTER

Already have an account?
Login

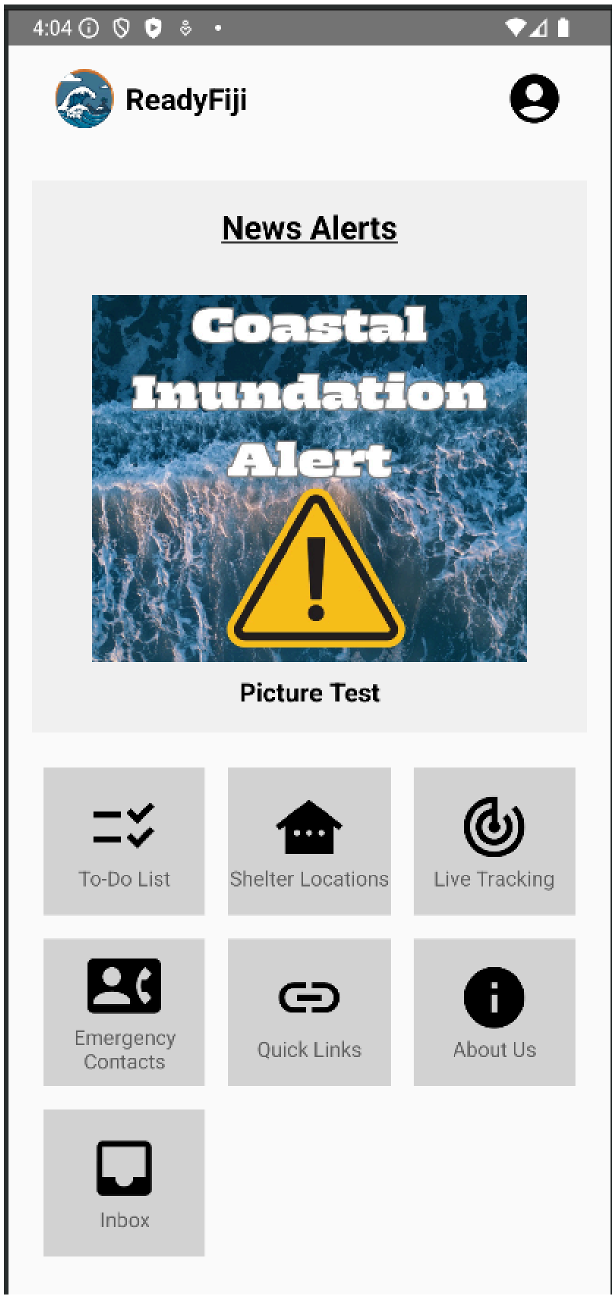
The screen above is a registration screen where users can create a personal account to access the features of the mobile app.

3.1.3 Login Screen



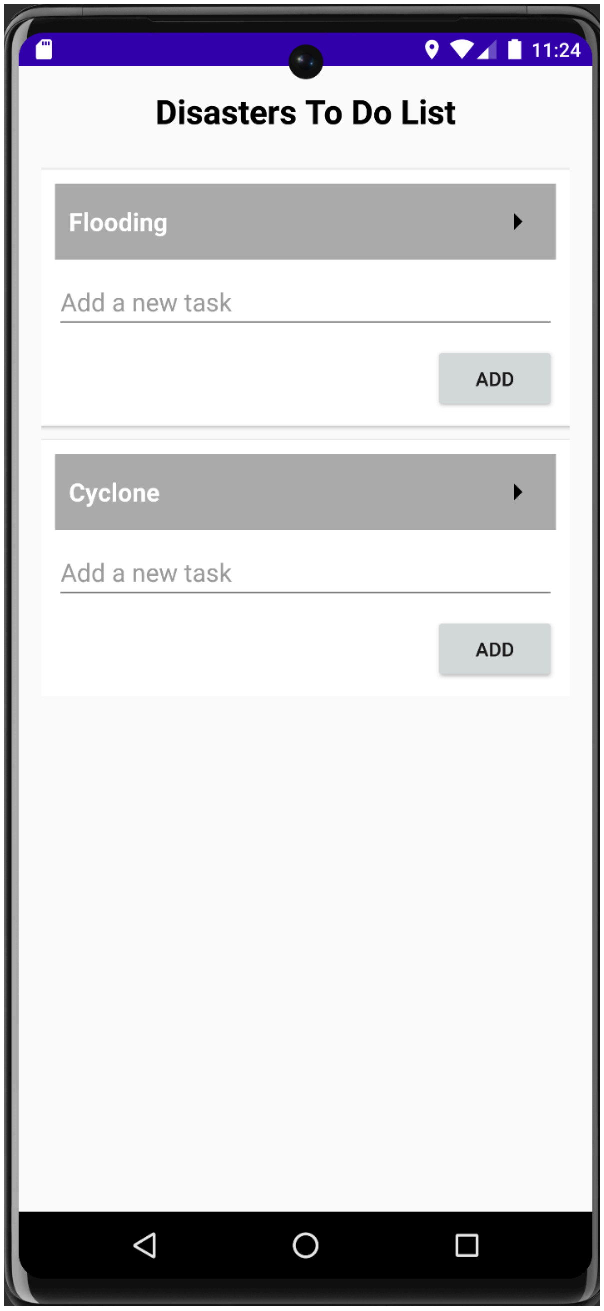
The screen above is a login screen where the user will need to authenticate themselves into the mobile app to access the apps services.

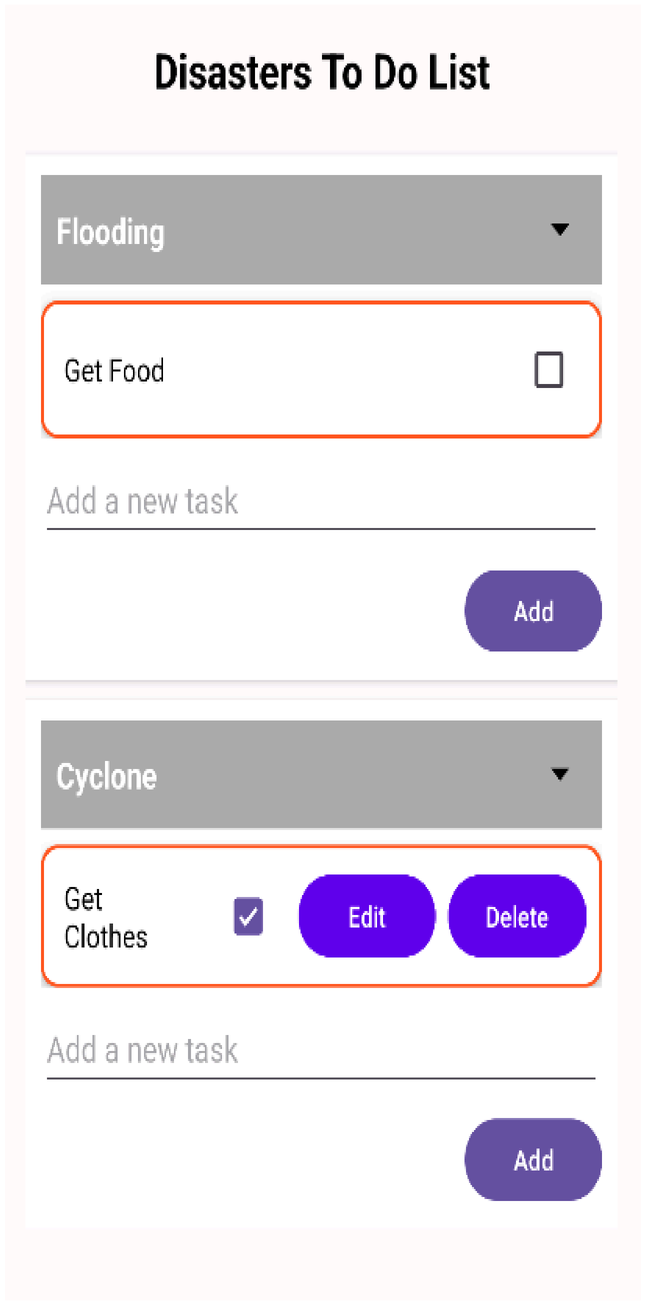
3.1.4 Home Screen



The following screen is the home screen where the users will be redirected to after they authenticate into the app. This screen contains buttons that will direct the user to other screens.

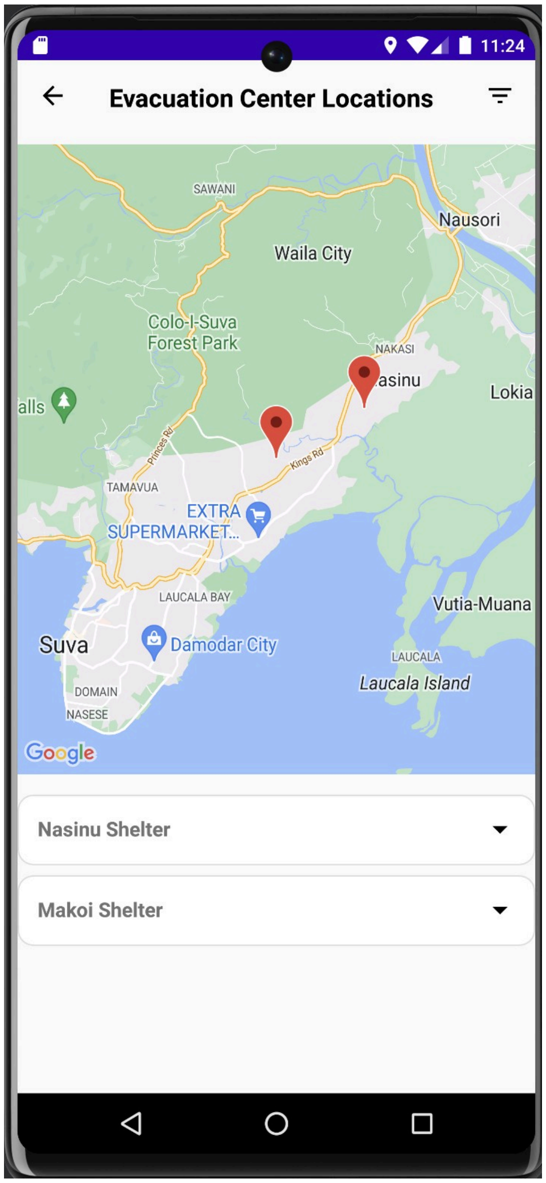
3.1.5 ToDo List and Disaster Tips Screen

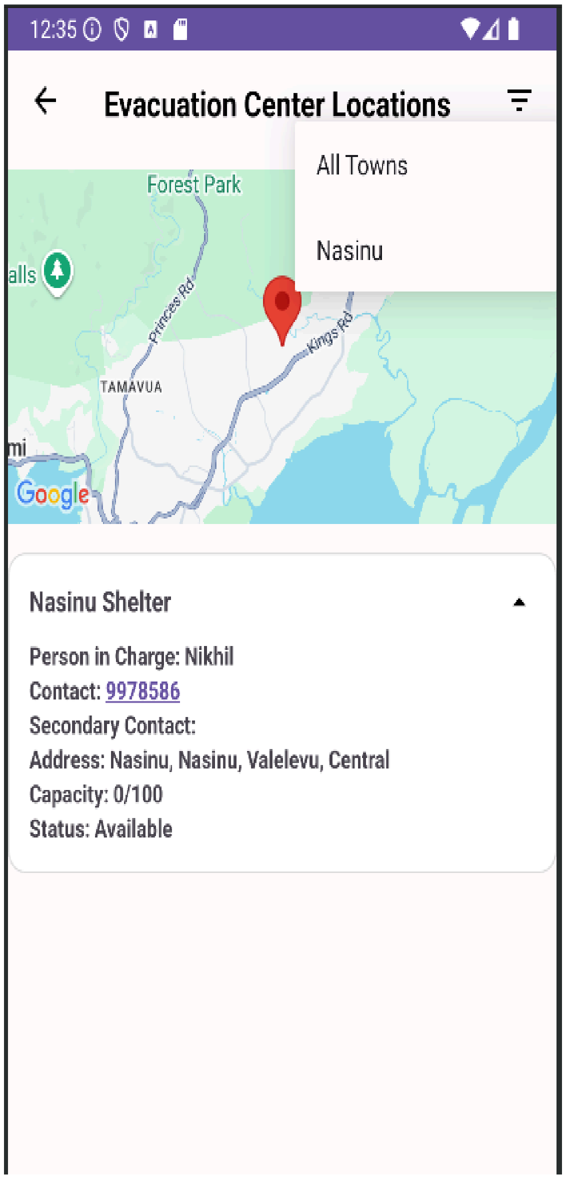




This screen is a disaster and supply tips screen. Here the user can get information from the server side where the admin will enter data for all users. This screen also allows users to create their own notes that may go beyond the information provided by the team of admins. This extra information could be user specific, that the users deems to be important.

3.1.6 Relief Shelter Location



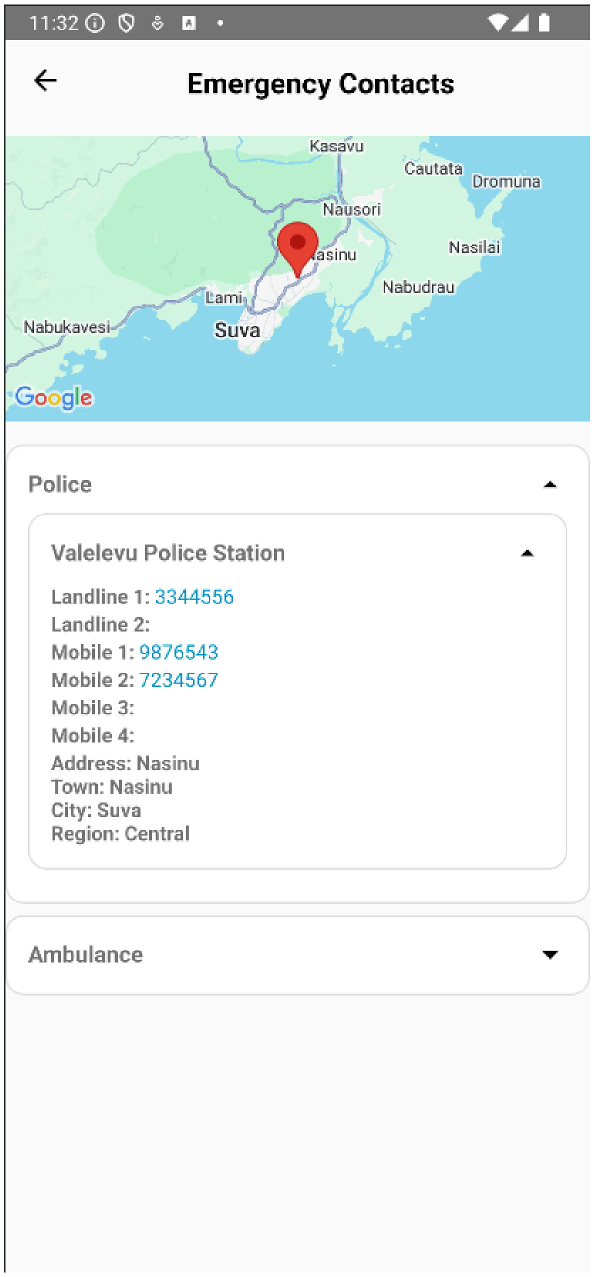


Nasinu Shelter

Person in Charge: Nikhil
Contact: 9978586
Secondary Contact:
Address: Nasinu, Nasinu, Valelevu, Central
Capacity: 0/100
Status: Available

The relief shelter tracking screen allows users to view shelters nearby to their location so that they can move to it in case of any danger or emergencies. Moreover this screen also shows the given description of a particular shelter.

3.1.7 Emergency Contact Screen



Emergency Contacts

Police

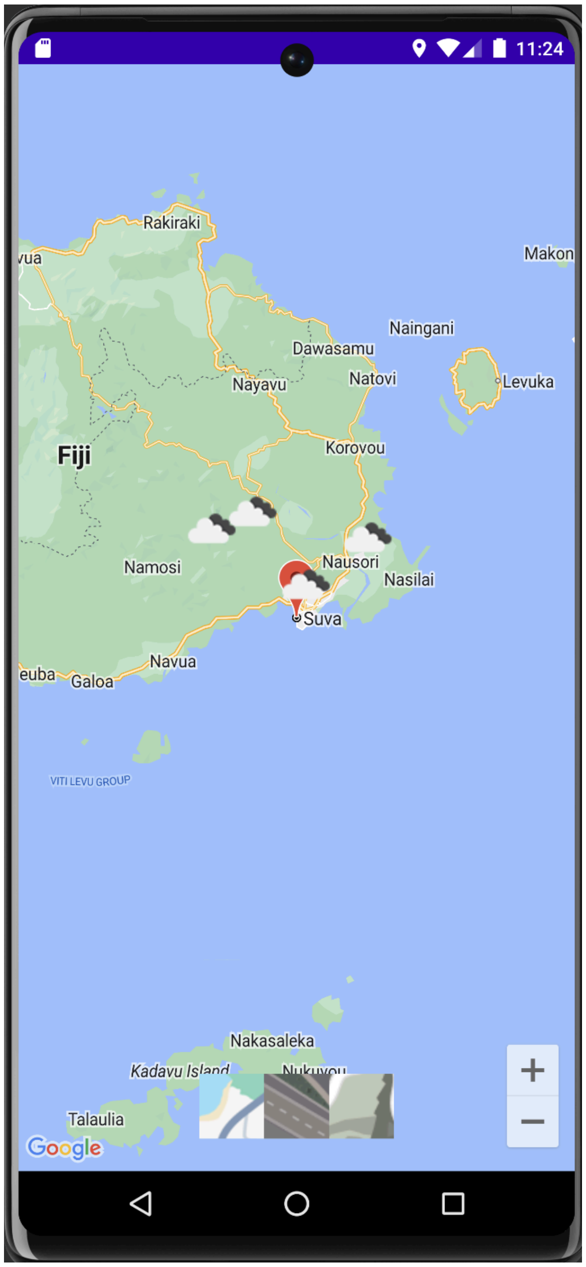
Valelevu Police Station

Landline 1: 3344556
Landline 2:
Mobile 1: 9876543
Mobile 2: 7234567
Mobile 3:
Mobile 4:
Address: Nasinu
Town: Nasinu
City: Suva
Region: Central

Ambulance

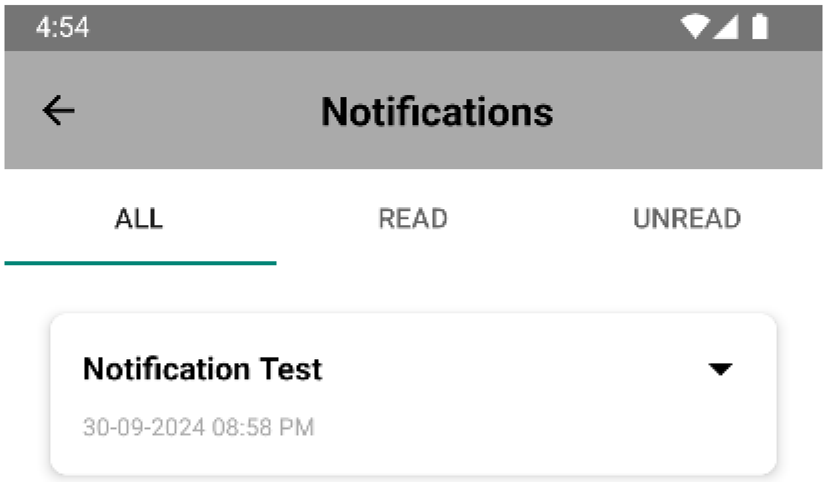
This particular screen shows locations of emergency institutions and services, and the emergency contacts for these services.

3.1.8 Live Disaster Tracking



This disaster map screen shows the live location of the disasters that may be occuring in your area.

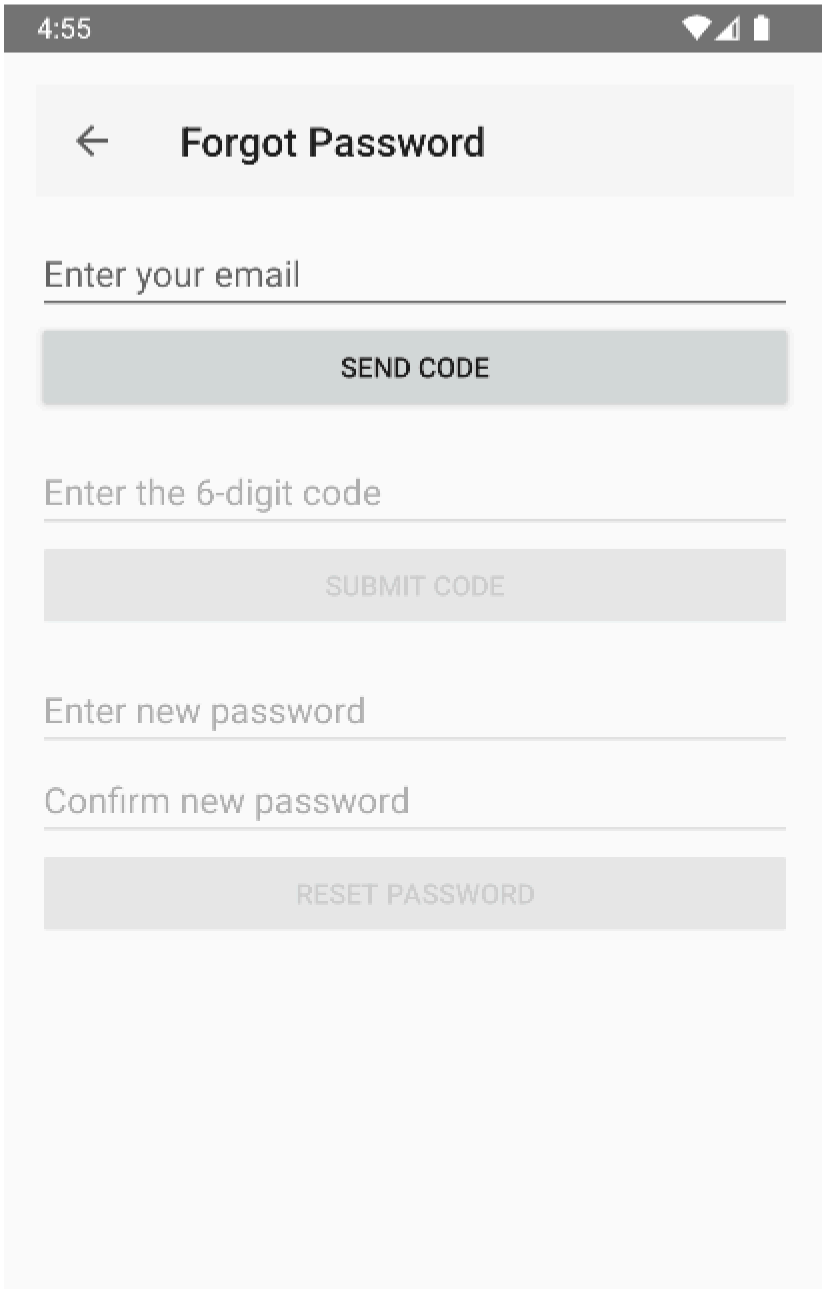
3.1.9 Notification tray



Notification Test
30-09-2024 08:58 PM

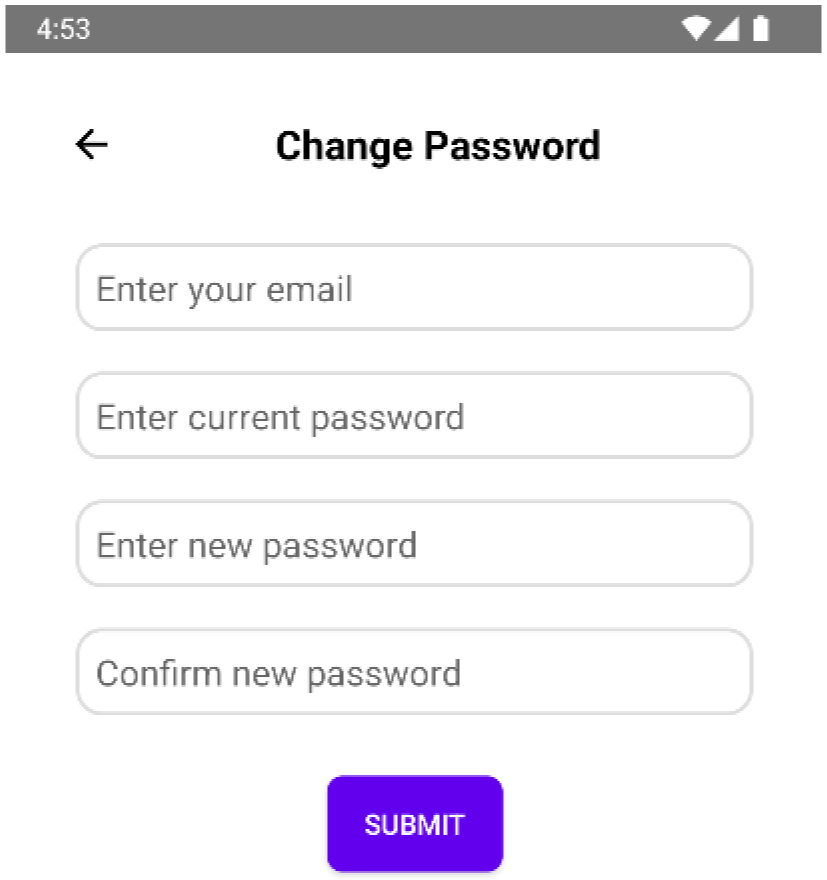
The screen above is a notification tray screen where all the alerts that the admin will make regarding the development of the disaster will be displayed.

3.1.10 Forgot Password



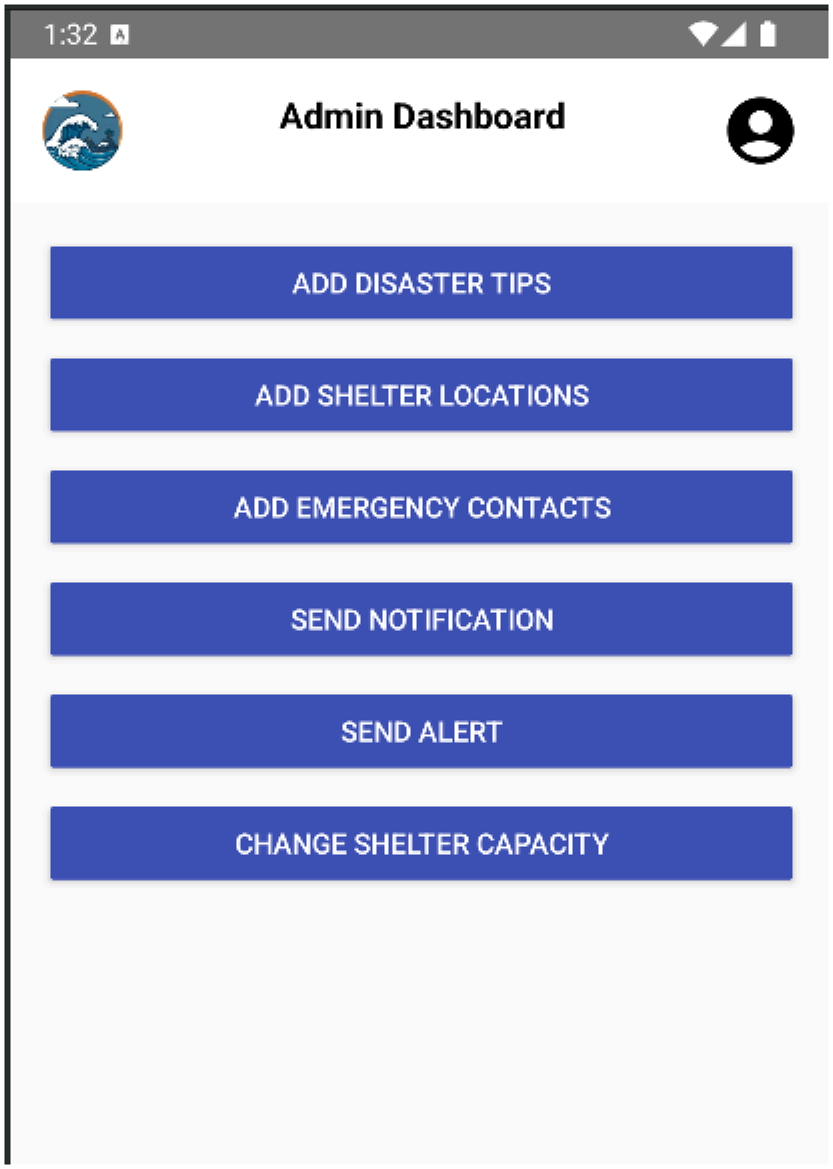
The forgot password screen allows users to create a new password if they have forgotten their old password to authenticate into the app.

3.1.11 Change Password



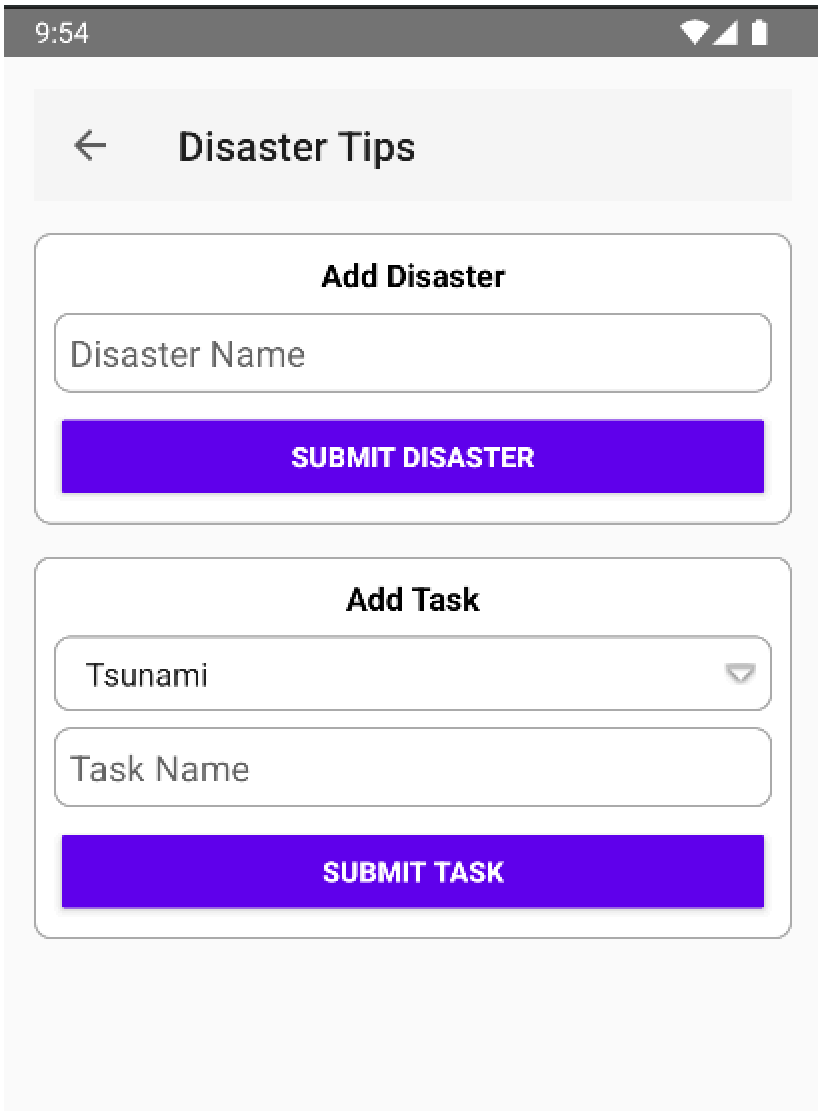
The change password screen allows users to change their password when they wish to change it into something else.

3.1.12 Admin Menu Screen



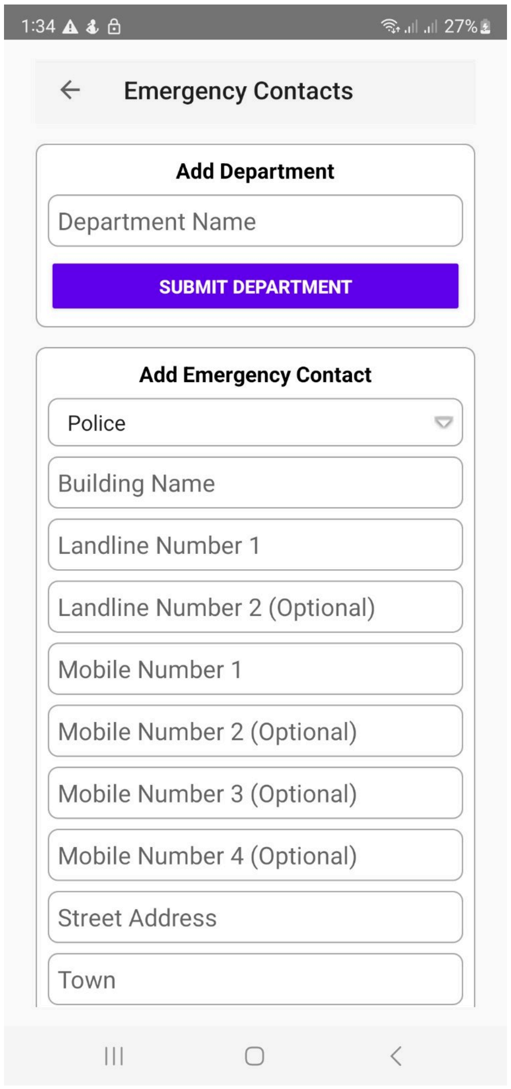
The above screen is the menu screen for admin users where they have a variety of options to different admin level functionality.

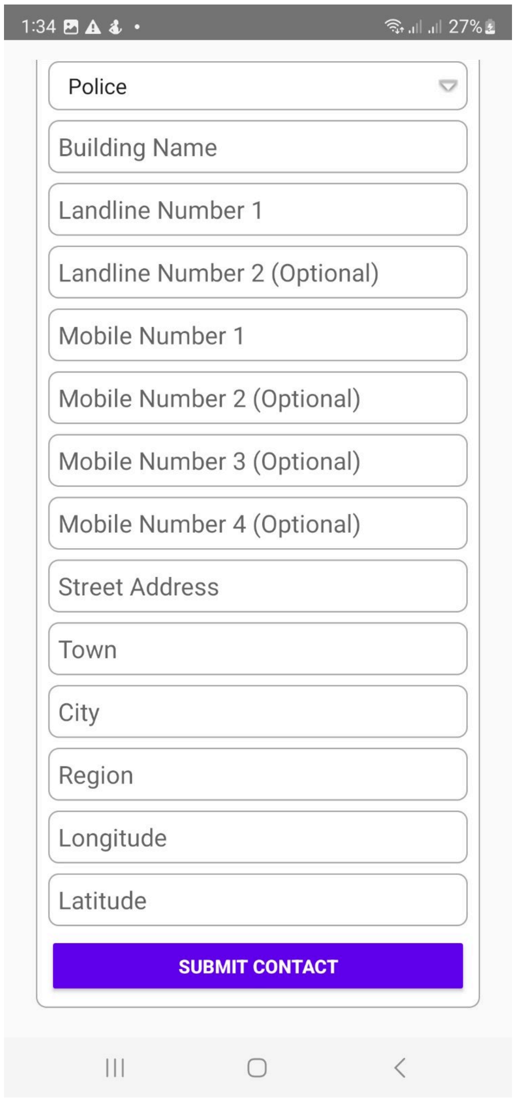
3.1.13 Create Disaster Tips (Admin Account)



The screen above is the admin user screen where the admin can create and add new disaster specific tips to the database. The user can then access the database to retrieve the information.

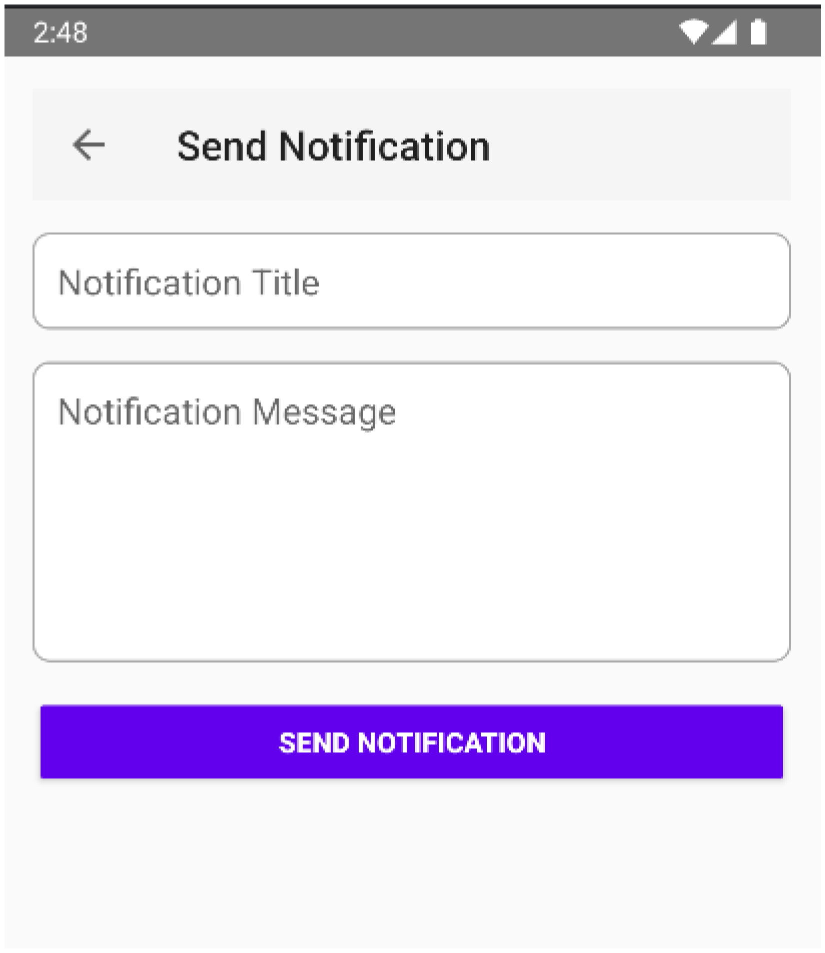
3.1.14 Add Emergency Contact (Admin Account)





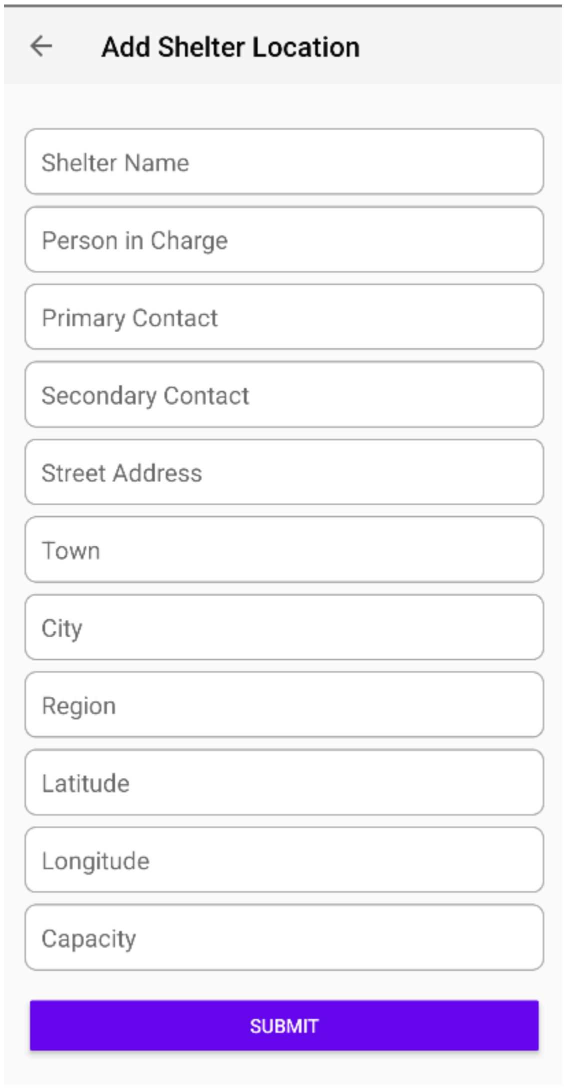
The above screen allows admins to add contacts of emergency services and display their locations on Google Maps. This allows users to always have contacts of various emergency services.

3.1.15 Create Disaster Alert (Admin Account)



The following screen allows the admin to create a disaster alert and send it to the users mobile devices. Users will be able to get up to date information on the progress of the disaster.

3.1.16 Create Relief Shelter (Admin Account)



This screen allows the admin user to create new relief shelters by adding the necessary details about it and saving it to the database.

3.2 Hardware Interfaces

Section 2.4 within this document highlights the requirements of an Android device where the application is to be downloaded. Any given device must match the listed requirements to gain maximum benefit.

The application has a push notification feature which requires the mobile devices to have a mobile network connection for the feature to function as intended.

The Disaster Preparedness app is a mobile app and therefore depends on Touch Screen, which allows multi-touch gestures as input. Moreover, the system, more specifically the mobile app displays all its content using a Graphical User Interface and thus requires a display to provide information to its users. The app also has a feature that allows it to push disaster alert notifications to the user's mobile's push notification tray. Most phones have an alert sound when a notification arrives and thus needs to interact with the mobile system's sound control system. The mobile app will also locally store data that may be retrieved from the server either permanently on the secondary storage or cache it into the primary memory and call upon it if a piece of data is needed. The app is a client server system and therefore requires the mobile app to connect to the central server for certain functionality or information. This inturn requires the mobile app to access the WIFI system to access the server through a network. Lastly, the app incorporates mapping functionality from the google API and thus requires sensors, specifically geolocation sensors to figure out the users location to enhance the user experience.

3.3 Software Interfaces

The device that downloads and uses this application requires Android 8.0 (oreo) or higher to be installed. Moreover, this application uses Google's Firebase as a real-time database manager for the purpose of sending disaster alerts as they occur. Furthermore, the app uses google maps API to display shelter locations on google maps. Also, the app also uses Windy API, which gets raw weather forecast data for specific coordinates, to display real-time location of natural disasters such as cyclones and various other weather patterns.

3.4 Communications Interfaces

When a disaster takes form or weather patterns seem adverse, the system will use a push notification feature to send alerts to the users. However, this does require the users to maintain a stable internet connection.

The mobile app needs to connect to the server to access information and this service request and request response requires a TCP/IP, a set of communication protocol, to send a request for a service from the mobile to the server and receive a response from the server back to the mobile. Moreover, the system also accesses information from external systems using APIs and requires app architecture to implement HTTPS to securely send sensitive information to these external systems and receive sensitive information back.

4. System Features

4.1 Creation, Update and Deletion of Information Regarding Disaster Action

4.1.1 Description and Priority

The admin user can create, update and delete information such as safety tips or other relevant information regarding a particular natural disaster. A full action plan will be listed as well as a preparation plan for each type of disaster and recommended actions that need to be taken to maximize safety and minimize personal damages to properties as well. The priority rating for this feature is high as it is a core feature of the mobile system. System should have disaster preparation and an action plan always ready for its users.

4.1.2 Stimulus/Response Sequences

- Admin users will enter information into a form on their interface, once completed the admin can save the information and commit it to the database.
- Data is saved to the database.
- When a user connects to the database, the new information is saved onto their mobile devices as the devices sync with the server.
- The end user can now access the new information on their mobile app.
- In the similar manner the admin can take existing information and change the content and save it to the server.
- The updated data is saved to the database.
- When a user connects to the database, the new information is saved onto their mobile devices as the devices sync with the server.
- The end user can now access the new information on their mobile app.
- Finally, the admin can delete the information stored on the server database.
- The change in data is recognised by the server.

4.1.3 Functional Requirements

- REQ-1: Admin can create information regarding disaster management and action and save it to a server that propagates to end users devices. Required admin level security privileges to access this service. (possible through admin authentication)
- REQ-2: Admin can update or delete information regarding disaster management and action and save it to a server that propagates to end users devices. Required admin level security privileges to access this service. (possible through admin authentication)

4.2 Creation, Update and Deletion of Relief Shelter Location

4.1.1 Description and Priority

The admin user can create, update and delete relief shelter locations to and from the database. The admin can set a new location of the relief shelter and give descriptive information about it, like what type of building it is and what facilities it has and how many people it can cater up to. In a similar manner, the admin can update existing relief shelters by changing its location, descriptions and facilities or delete them if they stop acting as relief shelters. The priority rating for this feature is high as it is a core feature of the mobile system. This is needed by users to see all possible relief locations in case they get displaced from their homes during a disaster.

4.1.2 Stimulus/Response Sequences

- When an admin wants to create a new relief shelter, the admin fills in a form that describes the location, description of building and facilities offered by a particular relief shelter.
- The location is then stored on a database on the server.
- As the end user’s mobile app connects to the server, the new relief shelters are added onto the device as device sync to the servers database.
- The admin can also update details about relief shelters stored in the database or delete them.
- The changes are stored into the database running on the server.
- Once the user connects their mobile app to the server and synchronizes with it, information is updated on the mobile devices local database.

4.1.3 Functional Requirements

- REQ-1: Admin can create information regarding relief shelters and save it to the database that propagates to end users devices. Required admin level security privileges to access this service. (possible through admin authentication).
- REQ-2: Admin can update or delete information regarding relief shelters and save it to the database that propagates to end users devices. Required admin level security privileges to access this service. (possible through admin authentication).

4.2 Push Disaster Alerts to mobile devices

4.1.1 Description and Priority

As the creation of a natural disaster becomes imminent, the admin can enable a message to be sent through the server to the end user's mobile device in real time. The message then is shown to the user through the push notification system of the android phone. This is a medium priority feature as it informs the users at the earliest of the development of the disaster rather than the user having to rely on other sources for information that might not be timely.

4.1.2 Stimulus/Response Sequences

- The admin creates an alert message that describes the situation and the best course of action.
- The alert message is then saved to the database.
- The server takes the alert message and relays it to the mobile app.
- The mobile app then displays the alert message in the push notification of the user’s mobile.

4.1.3 Functional Requirements

- REQ-1: Admin creates an alert message and saves it to the database on the server. Required admin level security privileges to access this service. (possible through admin authentication).
- REQ-2: the server relays the alert message to the mobile app.

- REQ-3: the mobile app then displays the alert messages content in the push notification tray. The Mobile app needs to be given permission to access the push notification trap or receive such a message when the app is on pause or has yet to be created.

4.2 Track Real Time Location of Disaster

4.1.1 Description and Priority

The mobile app connects to an external interface and displays a disaster map with the mobile application that depicts weather patterns and shows real time tracking of certain natural disasters. The priority of designing and implementing this feature is important as it's a self contained disaster tracking module that allows users to use their own discretion to act on the information from the weather map even before the admin sends out an alert message.

4.1.2 Stimulus/Response Sequences

- The mobile connects to the internet.
- The user open the mobile app
- The user accesses the disaster map and pinpoints their current location either manually or the app does it automatically using GPS.
- The user can then view and read the disaster map and if greater detail is needed, the user can open the website to view the full disaster and weather forecasting map.

4.1.3 Functional Requirements

- REQ-1: mobile app connects to the weather forecasting interface (this feature requires internet connectivity).
- REQ-2: user or app locates current location.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

A system performance requirements define how well the system functions under certain conditions. This requirement helps developers determine whether the system can continue to be responsive to user actions even after reaching the technically feasible boundary of operations set in place for the system. The following performance requirements are:

5.1.1 Response Time

The system should be able to access information from the server within 3 seconds and 10 seconds during peak user hours.

5.1.2 Efficiency

The system should allow up to 500 users concurrently to communicate with the server at any given time.

5.1.3 Resource Utilization

The mobile app should only utilize up to 200MB of RAM. In a similar manner, the system should not use more than 20% of CPU utilization.

5.1.4 Storage Capacity

The server should be able to store up to 10000 records within its database without crashing or slowing down.

5.1.5 Battery Consumption

The system should only consume no more than 5% of mobile's battery life for 1 hour of active use.

5.1.6 Error Recovery

The system should have an uptime of 98% during one operational period. In case of any error leading to a crash, the system should reset back to normal operating procedures within 5 seconds.

5.2 Safety Requirements

The following requirements are critical as they ensure no user of the mobile app comes in harm's way, whether to their own devices or to themselves. The mobile app accesses information or service from other external systems, however the accuracy and reliability of the information provided by these external services are not verifiable by the team of admins as these external services reserve all the rights for the information they provide. Henceforth, it would be recommended that users do not solely rely on the information from these external systems but use it as a guide to verify information they might receive from other sources. If any further enquiries are needed then users may consult the terms of agreement for each of the external service systems to identify the reliability of information these services provide.

5.3 Security Requirements

The Disaster Preparedness mobile app records users information and creates a user profile for each user of the app. This information is very sensitive as if it gets into the wrong hands can lead to identity theft and so much more. Therefore, the system is required to implement robust security features to protect illegal access to users personal data by encrypting data before transmitting over a network or storing into a database. Moreover, a new privacy framework should be drawn up to safeguard the information of the users and place limitations in what way the stakeholders of the system can utilize users data. Periodical system audits will enable system admins and maintainers to determine any major fault in the system or a potential breach in the security service of the system. Furthermore, any external system that may be implemented should ensure the services they provide do comply with the security measures of the mobile system. The system will communicate through the internet and the World Wide Web and therefore requires the Secure Socket License to ensure safe transmission of data through a network.

In addition, the end users and admin users both will be required to authenticate themselves into the system everytime they wish to access the services in the system. This is due to the security privilege level that each account type requires based on the service each account type facilitates.

5.4 Software Quality Attributes

5.4.1 Adaptability

The system should be able to run on multiple different versions of the Android operating System without compromising on functionality.

5.4.2 Availability

The system's services should be accessible 99% of the total time it is operational within a single operational period.

5.4.3 Correctness

The system should pass all of its test cases and operation with 100% accuracy and 0% system defects.

5.4.4 Flexibility

The system should be designed in such a way that further down its life cycle, additional modules can be added without having to reconfigure the entire system to add new modules and ensure existing one's run without errors.

5.4.5 Interoperability

The system should be able to integrate external systems and communicate with it error free.

5.4.6 Maintainability

The system's error detection and correction methodology should enable maintenance engineers to identify bugs in the system's service and correct it within 1 to 2 weeks from when it is first detected.

5.4.7 Reliability

The system should be able to provide service continuously for 11000 hours without any errors or crashes. Incase of any severe error, the system will log the error and switch to backup system protocol and restart the auxiliary server within 5 seconds of the error occurring.

5.4.8 Robustness

The system should be able to provide service at its technical operation limit and somewhat beyond without crashing or freezing. In this manner, any occurrence of a system error should be handled by the system itself.

5.4.9 Usability

The users of the system should be able to fully utilize their features and services of the system within an hour of self training through user guides or user manuals. Optionally, videographic tutorials could be designed for easier user training which may lower training time to below 30 minutes.

5.5 Business Rules

Admin users have the ability to set new information about disaster safety and other relevant tips. Admin has the ability to update or delete information about disaster safety. Furthermore, Admin users have the ability to set a new relief shelter location and description. They also have the ability to update or delete information regarding relief shelters. Admins can send an alert message to the mobile app that will be shown through the push notification tray.

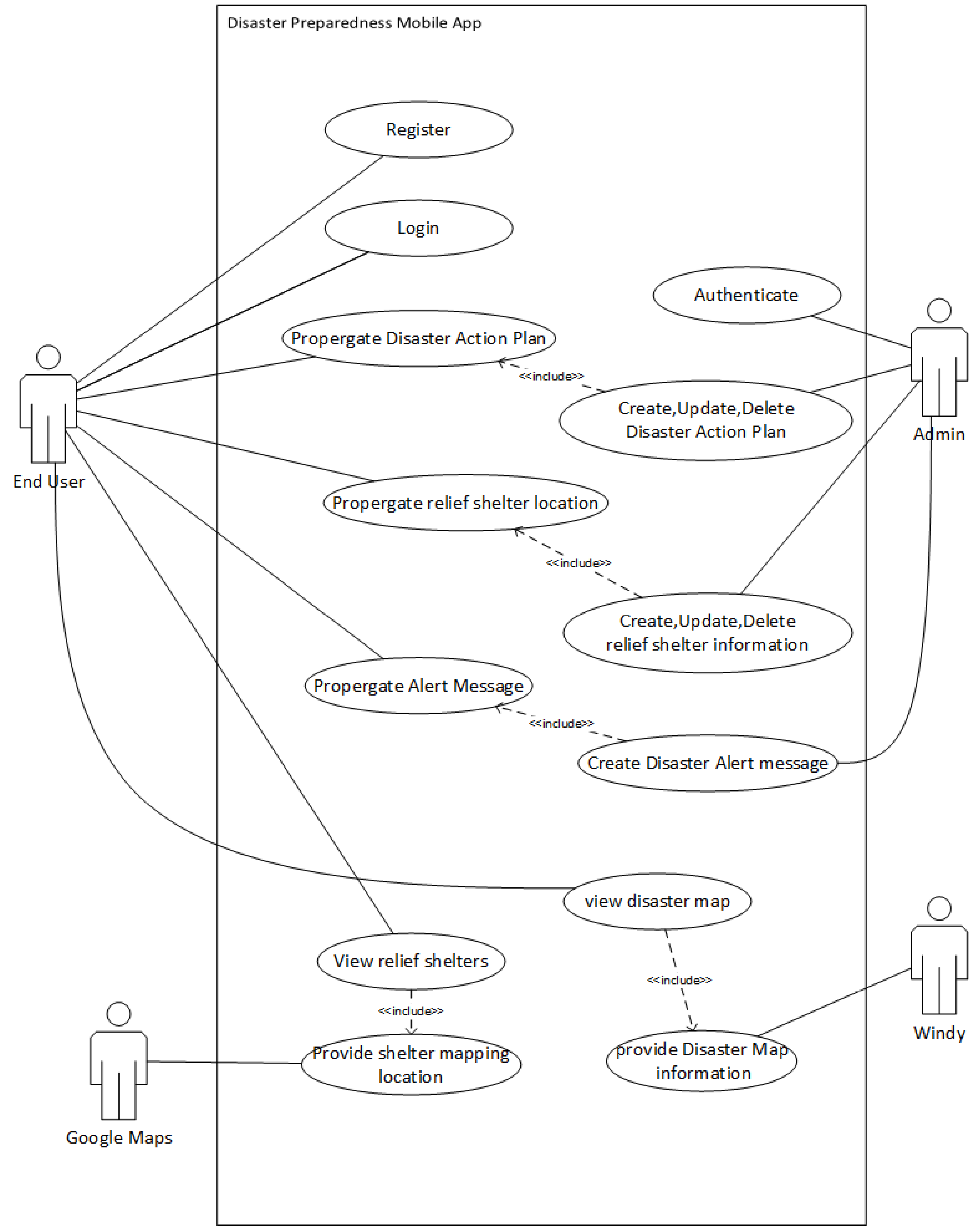
The user will receive all the following information from the admin user and will be able to view it within their mobile app.

6. Other Requirements

Appendix A: Glossary

Term	Definition
Windy API	A weather forecasting service whose features will be embedded into the system
Firebase API	A real time database management system that also includes messaging system
Google Maps API	A map system that locate points of interest on a map
API	A connection between to distinct systems that handle data in different formats
UML	Unified Modeling Language
QA	Quality Assurance
HTTPS	Hypertext Transfer Protocol Secure
TCP/IP	Network communication paradigm

Appendix B: Analysis Models


Disaster Preparedness Mobile App
Register
Login
Authenticate
Propergate Disaster Action Plan
<<include>>
Create,Update,Delete Disaster Action Plan
Propergate relief shelter location
<<include>>
Create,Update,Delete relief shelter information
Propergate Alert Message
<<include>>
Create Disaster Alert message
view disaster map
<<include>>
View relief shelters
<<include>>
Provide shelter mapping location
provide Disaster Map information
End User
Admin
Windy
Google Maps

Appendix C: References

Are Nonfunctional Requirements? Definition + Best Practices. (2022, June 12). Jama Software. Retrieved October 2, 2024, from https://www.jamasoftware.com/requirements-management-guide/writing-requirements/how-non-functional-requirements-impact-product-development

Firebase. (2023, May 12). *SDKs and client libraries | Firestore | Firebase*. Firebase. Retrieved September 26, 2024, from https://firebase.google.com/docs/firestore/client/libraries

Jain, S. (2024, June 27). *How to GET Data From API using Retrofit Library in Android?* GeeksforGeeks. Retrieved September 26, 2024, from https://www.geeksforgeeks.org/how-to-get-data-from-api-using-retrofit-library-in-android/

Krüger, G. (2024, August 8). *Non-functional Requirements: What They Do, Examples, and Best Practices*. Perforce Software. Retrieved October 2, 2024, from https://www.perforce.com/blog/alm/what-are-non-functional-requirements-examples

Krüger, G., & Lane, C. (2023, January 17). *How to Write an SRS (Software Requirements Specification Document)*. Perforce Software. Retrieved October 2, 2024, from https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document