



Final Year Dissertation

The Effectiveness of Personal Safety Applications

Jamie McCulloch

Heriot Watt University

H00189648

jm7@hw.ac.uk

MEng Software Engineering

Supervisor: Mike Just

Second Reader: Sven-Bodo Scholz

Acknowledgements

I would like to thank Mike Just for suggesting this topic, and for all his support and advice throughout this project. Also, Sven-Bodo Scholz for his advice as second reader.

Declaration

I, Jamie McCulloch, confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of other authors in any form (e.g., ideas, equations, gs, text, tables, programs) are properly acknowledged at any point of their use. A list of references employed is included.

Signed:

Date:

Abstract

As smartphone ownership continues to rise, they have become an integral part of society. Their capabilities and services are used daily, and continue to grow. The ability to track locations, contact relatives and emergency services within seconds, and instantly record video and audio has ensured that the general public can use their smartphones to improve their personal safety. However, this avenue has not been well explored, and a question mark resides over this - are smartphones able to improve personal safety? To find this out I have designed and implemented a personal safety application called KeepSafe for the Android platform, and carried out multiple studies to gauge societal opinion on if this improved their perception of personal safety. On the whole, the majority of participants in both studies felt that this would benefit their life, and they would feel safer using a personal safety application, specifically KeepSafe.

Contents

1	Introduction	7
1.1	Overview	7
1.2	Aims and Objectives	7
1.3	Challenges and Possible Solutions	8
1.4	Layout of Dissertation	9
2	Background	10
2.1	Gender Difference	10
2.2	Age Difference	10
2.3	Need for Personal Safety Applications	11
2.4	Problems with Personal Safety Applications	12
2.5	Perception of Smartphones	13
2.6	Characteristics of Feeling Unsafe	13
2.7	Mobile Phone Safety	14
2.8	Related tools	14
2.8.1	Social Media	14
2.8.2	Existing Applications	15
2.9	Android	16
3	Requirements Analysis	17
3.1	Survey Results	17
3.2	Functional Requirements	19
3.3	Non-functional Requirements	20
4	Project Design	21
4.1	Use Cases	21
4.2	Initial User Interface and Name	22
4.3	System Architecture	23

5	Implementation	24
5.1	Intended functionality	24
5.2	Back End	25
5.3	Prototype 1	28
5.3.1	Implementing Prototype 1	28
5.3.2	Evaluation of Prototype 1	30
5.4	Prototype 2	31
5.4.1	Implementing Prototype 2	31
5.4.2	Evaluation of Prototype 2	33
5.5	Prototype 3	34
5.5.1	Implementation of Prototype 3	34
5.5.2	Evaluation of Prototype 3	35
5.6	Unique Features of KeepSafe	35
5.6.1	Panic Button	35
5.6.2	Alert Mode	38
5.6.3	Journey Countdown	39
6	Components of KeepSafe	41
6.1	Front End Components	42
6.2	Back End Components	43
7	Testing	44
7.1	Unit Testing	44
7.2	Component Testing	44
7.3	Usability Testing	44
8	Usability study	46
8.1	Methodology	46
8.2	Evaluating Usability Study	48
8.2.1	Initial Thoughts of Application	48
8.2.2	Adding a Contact	48
8.2.3	Triggering Panic Alarm	48
8.2.4	Configuring Panic Button Settings	49
8.2.5	Sending a Location Update	50
8.2.6	Phoning emergency contacts	51
8.2.7	Activating Alert Mode	51
8.2.8	Journey Countdown	52

8.2.9	Final Thoughts on Application	52
8.2.10	Age Difference	52
8.2.11	Conclusion of Study	53
9	Conclusion	54
9.1	Achievements	54
9.2	Limitations	54
9.3	Future Plans	55
9.3.1	Time Filters	55
9.3.2	Proximity Based Social-Media	55
9.3.3	Fake Call Feature	56
9.3.4	Alternate Route Planner	57
A	Survey Questions	61
B	Survey Results	63
C	Use Cases	65
C.1	SendLocation	65
C.2	addContact	66
C.3	firstOpen	67
C.4	setJourneyCountdown	67
C.5	PanicButton	68
C.6	AccessingSocialMedia	68
C.7	PostComment	69
D	Consent Form	71
E	Usability Questions	72

Chapter 1

Introduction

1.1 Overview

Smartphone ownership is at an all time high, and it is imperative that they are used to their full potential. With this in mind, one area in which there is not a lot of research or evaluation into is personal safety. Smartphones are able to do many things, such as:

- track GPS location
- store thousands of contacts
- record video and audio instantly
- access the internet through broadband cellular networks

It is now possible for people's location to be tracked, for others to be immediately contacted, and instant proof of crime through video/audio recording. The rise of social networking has allowed people to spread news, and their own personal experiences of certain areas within their local community, allowing people to gauge what surrounding areas are safe to walk or not.

1.2 Aims and Objectives

The aim of this project is to develop a smartphone application that improves the personal safety of its user while they are walking outside alone. This will be done through location updates and tracking, an emergency panic button, alternate route planner, among others. The objectives for this project are listed below:

- **Designing the application**

Designing a personal safety application with an intuitive user interface, that users can easily follow

and complete tasks with, and with features that people want. I have carried out an online survey, with 74 participants, asking their opinion of personal safety applications and if they would find them useful. The survey comprised of 9 questions, and a copy of the survey questions is available in Appendix A. I kept the survey online for 5 days, then closed it and sorted through the responses. I also used the survey to gather requirements for the application laid out in 3 of this document. I created use cases during the design stage of the project (available in Appendix C), as well as designing the user interface (see section 4.2), and the prototypes for this project (see section 5).

- **Building the application**

I have implemented my own personal safety application, using the research I've found. I implemented features from other applications (panic button, emergency contacts, location updates) but also intended some of my own ideas (a journey countdown timer, a proximity based social media, time filters). I created three prototypes, laid out in section 5, with incremental differences, resulting in a fully functional application that implements the most popular features laid out in section 3.2 of this report, as well as a few others.

- **Evaluating the application**

Once I created a prototype application, I then recruited a small group of participants and conducted an interview with them to get their feedback in terms of the application. I asked them to complete tasks on the application, and asked if they have any improvements they would make. More information on the usability study is found in section 8 of this report.

1.3 Challenges and Possible Solutions

The challenge with personal safety applications is how people determine somewhere being unsafe. For example, social media can be used to alert to unsafe locations, or unusual activity, but with the sheer scale of information on social media, and the privacy surrounding individual profiles, this information can easily get hidden or missed. The proximity based social media platform I hoped to integrate into my application may potentially solve this, as anyone will be able to post to it and only those in a certain area will be able to see posts.

There are many factors that determine if somewhere is unsafe - noise level, past crime rate, time of day, the people residing there, etc., [1] and these all have to be considered individually and refined.

People may not want to be able to be tracked, or want to know when they are entering an unsafe location, essentially, due to privacy concerns

I had hoped that my personal safety application would prevent these problems. It would extract information from local council/authority Facebook/Twitter feeds, and also rely on users contributing directly to the application, anonymously, their experiences. The application would run only when the user selects it, and all features are able to be turned off and on as the user pleases. While not in this implementation, in section 9.3 I discuss future plans in terms of the application.

1.4 Layout of Dissertation

Section 2: In this section I have conducted a literature review into the topic of personal safety, including differences in gender and age, problems with personal safety applications, and the need for a personal safety application, among others.

Section 3: Using my survey, I derived requirements, both functional and non functional, and conducted analysis on these.

Section 4: The design of the project, including an activity diagram, use cases, and the initial design aspects of the project.

Section 5: This section outlines the implementation of the prototypes, and evaluates them at each stage. Also outlines some of the unique and advanced features of the application.

Section 6: This section outlines the activities and back end files for KeepSafe.

Section 7: The initial testing I conducted on the application.

Section 8: Outlining the usability study I have conducted, evaluation the results, and how I would apply this to my application.

Section 9: This section outlines the achievements and limitations of the application, while also talking about future plans.

Chapter 2

Background

This section will investigate and evaluate existing research into the use of smartphones for safety applications.

2.1 Gender Difference

Males and Females have different habits when it comes to their smartphones. A study by the team behind 'uSafe' found that 72% of females (in their study) reported feeling unsafe while walking at night. This is compared to 62% in males [2]. Although only 10% of a difference this still shows that on the highest level, men either have lower levels of fear, or they're embarrassed to admit to feeling unsafe.

Men also rarely use their phone to increase their safety [3], with 75% in this study claiming to have never used their phone for safety. On the contrary, 65% of women in this study claimed to use their phone all the time to increase their personal safety.

According to the Bureau of Justice in 2009, females are more likely to be sexually assaulted than males [4], however more surprisingly, men are more likely to be assaulted or robbed than females. With this in mind, a personal safety application would be beneficial for both genders, contrary to popular belief, not just females.

2.2 Age Difference

Age is another factor that determines whether people would want/use a personal safety application on their phones. For example, in a group of 50-68 year olds, 81.7% sent less than 10 texts a day [5], compared to 22.5% of 18-24 year olds sending less than 10. This could be a range of factors, one of which being the fact that they may not feel as comfortable sending multiple texts to the same person in a day. This could be an issue in terms of the application, would they feel awkward sending their location to contacts if they thought they were in trouble for fear of being annoying?

However people would rather text than phone in every social situation other than driving (which is illegal in most districts) [5] which highlights that any application should favour a text based system rather than a phone based one, which I will look into for my own application.

The younger generation, between 18-34 showed high signs of anxiousness if they were to lose their phone, displaying a high emotional attachment to it [5]. This goes to show that people tend to have their phones on them at all times, and rarely go without, meaning the application would be highly accessible, and available.

Younger women are more likely to contact their parents more, in general, than males, this also hints to the fact that women are more likely to want someone to know where they are and use a personal safety application [5]. Charlton *et al* found that 38% of children aged 10-11 had used their phone in a crisis situation [6]. The younger generation who are growing up in the smartphone generation are using them effectively to enhance their personal safety. There is no description as to what these situations were but with the percentage being so high, its clear that the phones have been advantageous for these children.

2.3 Need for Personal Safety Applications

Smartphones are becoming increasingly popular, and accessible, with smartphone penetration rising by 33% in the last 5 years [7] in the UK, it is clear that most people have them. These can be incredibly useful when confronted with a crime, or when trying to avoid an unsafe or uncomfortable situation. The same can be said around the world, smartphones are on the rise, and show no signs of stopping. With features like GPS tracking and the improvement in carrier coverage, people can be found and contacted easier than ever before. With this in mind, generally the amount of crimes being committed is lowering [4], whether or not this is due to the rise of smartphones is unknown, however it still happens thousands of times a day, and may sometimes be avoidable.

There was a large incident in Dehli, India in 2012 which involved two women being brutally attacked [8] and raped on a public bus. Police recorded more than 550 cases of rape in Dehli in 2011 [9], leading to Dehli being called the “rape capital of India”. To counter this, the Indian government have mandated that all mobile phones must have a panic button installed in 2017, and all phones must have GPS by 2018 [10]. Considering the popularity of smartphones, and the constant threat of these sorts of crimes happening, it seems only logical to combine these and prevent crimes of this nature happening as much as they have been.

Crime can sometimes be avoidable. In some cases it definitely cannot be, but there are some methods to avoid it. For example, avoiding a certain area that you know is prone to criminal activity or is poorly lit. You can also (pretend to) be on the phone, which may deter some criminals as there is someone who would instantly know that you are in danger. Most of these are common sense and easy to do, but can

be incredibly useful when someone is feeling unsafe. A personal safety application can combine all of these ideas into one place.

2.4 Problems with Personal Safety Applications

In this current society, privacy concerns are an all time high. The thought of being constantly tracked and easily located is scary to most. Snapchat recently implemented a location tracking feature, which instantly raised fears of stalking and bullying [11]. Although it is intended to be a way of inspiring adventure and spontaneity, a large percentage saw it as a negative feature. It is essential to a personal safety application however, and arguably people would expect it to be there when using the application. It will, as well as all other features on my application, be optional.

The uSafe study found that 94% of its participants said that it was important for their privacy to be respected while using the application, and 90% would prefer to stay anonymous when using the application [2]. A problem that arises from this would be that once something is anonymous, it can be difficult to track what information is correct and what is not. As there is no way to tell who is contributing information, it becomes unreliable. Anyone can potentially add data to the application, without filter and without trace, skewing results. To combat this, there could be local moderators, to regulate the information being posted, while on the implementation side, it may be possible to introduce a filter, to regulate swearing and insults, for example.

It was found that a quarter of people in the uSafe study claimed to not appreciate using a personal safety application [2], and 45% would not want to be informed when entering an unsafe area. Furthermore, 22% strongly disagreed to being informed [2]. There may be a lot of reasons attributed to this. For example, the user may follow an *ignorance is bliss* motto to life, and if the application was to tell them that they were entering an unsafe area they may become paranoid and feel uncomfortable for no reason other than someone has previously reported that it is a dangerous area. It may also be due to the user thinking that they are being constantly tracked, and that every step they take is monitored. This would not be the case with my application, as it would need to be opened by the user and exits as soon as the home button is pressed. However it is still a rational fear, especially in current society where everything is monitored.

The user may also not want any unnecessary notifications on their phone, as they can be annoying and overbearing if going off consistently.

Alerts from a personal safety application may only inform the user of a dangerous event or unsafe place once it has happened or once the user is in the unsafe area [12]. This would need to be overcome as it

essentially renders the application pointless.

2.5 Perception of Smartphones

While features of smartphones are designed to improve the daily lives of its user, making everything as simple and accessible as possible, with that comes the risk of people using them inappropriately. A study into primary school children by Chartlon *et al* [6] found that 14% of primary school children had admitted to sending an inappropriate/threatening message to someone else, and 17% had admitted to receiving one. Cyberbullying is becoming increasingly more common among school children, both primary and secondary. A study found that 34% of students admitted to being cyberbullied [13], with 40.6% of females being cyberbullied compared to 28.2% in males. This is in part due to the increase in social media, in 2015 it was found 92% of teenagers aged 13-17 were online everyday and 24% were online *constantly*. With smartphones enabling internet access anywhere with 4G it is increasingly easy to get online. Due to this a lot of the older generation view smartphones negatively, as a way of interfering with children's lives and causing them undue stress, anxiety, and depression.

On the other side, it was found that improving personal safety was one of the main reasons for teenagers getting a phone [14]. It allowed young people to be in contact with their parents/guardians if they need them and gave the parents/guardians a sense of comfort knowing that their child was only a phone call away. Despite the cyberbullying aspect, smartphones are an incredibly useful tool for the younger generation. If they get lost, or need help it is easy for them to phone for help, or to load up a map-based application and find a way to wherever they are going.

Owning a smartphone can be a double-edged sword. It may help prevent a crime, i.e. being on the phone may deter a criminal, or the phone may be used if someone is lost, instead of asking strangers or walking around aimlessly in an unknown part of a city. However if someone is walking around the streets with an expensive smartphone, a criminal may attempt to steal the phone. Theft of smartphones makes up for a third of all personal robberies in the UK [3], meaning it is a common occurrence. This was highlighted in a study by Downes and Aoki when participants were asked if a smartphone helps or hinders their personal safety [14].

2.6 Characteristics of Feeling Unsafe

Although somewhat obvious, it is helpful, and useful, to be able to characterise factors which determine whether someone is feeling unsafe. For example, poor lighting is a significant factor in determining someone's safety levels. It was found that more lighting would improve personal safety [1], along with CCTV cameras when it is dark.

Another common characteristic to feeling unsafe is the visibility of people around you. If it is dark, and another person (or group of people) is visible, it can improve the feeling of safety. If anything happens there will be witnesses and help [1]. However, there is also the possibility that it is detrimental to feeling safe. It may be that some feel safer thinking they are alone, if there is no one around, then there is no one to attack them. This is subjective, it differs from person to person and there is no pattern to determine this.

People feel significantly more unsafe if they are walking in an unfamiliar place, or in an area that is notorious for having criminal activity. While this seems obvious, it is important as there may be occasions in which these circumstances are unavoidable. It may be that the shortest route home travels through an unfamiliar place, or an unsafe place. Nasar found that females are more likely to take the shortest route home, with 71.6% preferring it to 46.0% of males [15]. Any personal safety application would have to bare this in mind when planning any route to a destination.

2.7 Mobile Phone Safety

Mobile phones have long been used for personal safety, even before smartphones were the phenomenon they are today. Police advise the public to phone friends when walking alone, as it makes them look less vulnerable. It also deters criminals, as they are aware that someone would be instantly alerted if they were attacked [1].

Mobile phones could also be used to improve safety as the user knows that they can contact the emergency services, or a friend or relative if they are in trouble. Knowing this, even if it is not being used, can take a weight of one's mind and make them feel safer [16].

Just having a phone in their possession can improve someone's sense of safety, it was found that the majority of undergraduate students feel safer when walking alone if they have a mobile phone on their person. As mentioned above, just the feeling of knowing you can contact someone in an emergency and that you have a way of being contacted [17].

2.8 Related tools

2.8.1 Social Media

In today's society, smartphones are everywhere. A large percentage of people have one, and use it frequently. Smartphones can be used for many things, but one of the most common aspects is social media. With phone contracts becoming more common, and each offering high amounts of 4G data, the ability to check in and share your day with friends is easier than ever. Social media refers to "websites and applications that enable users to create and share content or to participate in social networking."

according to the dictionary definition. Although these were predominantly used by younger age groups, more recently older age groups have started using it as a way of finding old friends and keeping up with family. Social media has existed for years, from sites like MySpace, Friends Reunited, and Bebo all the way to current sites like Facebook, LinkedIn and Twitter. There is a plethora of information on social media, some useful and some not, and this is one of its biggest downfalls - it is difficult to find information once it's more than a few hours old. Posting about crimes and unsafe situations has become more popular in recent months. It is a way for people to share their experience with their local friends, warning them of dangers and of areas that they've felt unsafe, as well as if a crime has been committed - i.e. car broken into. Trying to find this sort of information within the labyrinth of social media is difficult. With a personal safety application, this would be the only information featured, and so would be easy to find and relate to.

Each local district, at least within Scotland, has a local police division page on Twitter and Facebook, which they use to alert people of crimes or dangerous areas around the local community. This is an essential tool when personal safety is involved, and it would be useful to integrate these together in the application. Not all the information on these pages is pertinent to crimes however, there tends to be fundraising activities, or updates to the local police force as well.

Each social media account can have a lot of personal information on it, and a lot of people have concerns over this data. Regulating a profile to only show posts to 'friends' is simple, and is commonplace. This means that a lot of information is hidden, only to be viewed if you are already connected to someone. A lot of people locally may not be friends with each other, or follow each other, meaning their warnings go unread by people that would find it most useful.

2.8.2 Existing Applications

A number of personal safety applications exist for both Android and iOS, including StaySafe [18], bSafe [19], and SafetiPin [20]. All of these have their own special features and unique selling points. For example bSafe has a "Follow Me" feature which allows contacts to watch your journey on a map, tracking your movements to ensure you don't stray from your path. This feature is incredibly useful, it instantly allows your trusted contacts to see where you are, and if you deviate from a path or stop walking then it is clear that something is wrong. Although features like this may sometimes cause undue panic (if the user stops to talk to someone they know, or goes another way home) however this is easily remedied, just by texting them or phoning the user.

There is very little research being done into the effectiveness of these applications, and if they are a solution to the larger problem of personal safety.

Google has developed applications that allow proximity-based alarms based on the user's location. Google Keep [21] is a reminder based application that alerts the user to alerts that they have previously set. The interesting thing is that while it can be a time-based reminder which is commonplace in reminder applications, it can also be location-based, allowing you to set a reminder of a shopping list, for example, and it would remind you once it detected your location as being at a supermarket. While this is not directly related to personal safety, the same technology can be applied to a personal safety application. For example, if an area has been flagged as unsafe by multiple reports and a user goes into this area, then the application may alert them and inform them they are in an unsafe area. As mentioned in Section 2.4, it was found that 45% of participants in the uSafe study would not like to be informed if they enter an unsafe area [2], although this also signifies that 55% would appreciate this feature, or at least are neutral towards the idea of it.

2.9 Android

Android is an operating system, developed by Google and utilising a modified version of the Linux kernel as well as other open source software. The first public release of Android was in 2008, with a very simple interface. This however over the years has been constantly updated, and optimised to the point it is by a significant amount, the most popular operating system on smartphones today. Android holds 75% of the market share among smartphones in February 2018 [22]. I am implementing this application on Android, as this means there is a higher percentage of people able to use it, and the application is there to try to improve personal safety - the more users the better.

Chapter 3

Requirements Analysis

3.1 Survey Results

For my initial survey, I used SurveyPlanet to create and share. The survey consisted of 9 questions, mostly multiple choice, and was geared towards gathering requirements for the application as well as gauging the attitude towards personal safety applications in society. 74 participants responded to the survey, ranging from 18 years old to 70 years old, and 26 male respondents (35%) compared to 48 female (65%). This gave a significant variation in results and allowed me to see the trends among age groups and gender difference, and see if they match with the studies already discussed in sections 2.2 and 2.1. Below are the results of the most important features:

Feature	Count
Location Tracker/Updates	61
Panic Button	55
Alarm Sound	39
Fake Incoming Call	30
Alternate Route Planner	15
None	3
Selfie Mode	0

Through this survey, I also tried to gauge what the user is looking for in a personal safety application outwith functionality. I asked questions related to feeling unsafe based on the time of day, or the situation they are in. I found that 77% of participants felt more unsafe between the hours of 6pm and 6am. This was expected, as this is when it tends to get darker outside and less busy, and people often associate this with feeling unsafe. I also discovered that 82% have felt unsafe while walking outside. This shows that an application specialising in personal safety can be very beneficial in society, and that there would be a demand for this. With the majority of people owning smartphones, and Android holding a huge market

share of these phones, as discussed in section 2.9, a personal safety application on Android would be highly accessible to everyone.

One of the questions from my survey asked for an age range of the participant. Personal safety is not simply just aimed at people walking at night, it is for everyone. Within my study, 52% were over the age of 25, and 24% were over the age of 56. Smartphones are no longer for the younger generation, they are accessible to everyone, and through this survey I can deduce that a personal safety application would be just as helpful for the older generation as it would be for the young generation. This means my application would need to be easy to understand, intuitive to use, and simple to navigate.

3.2 Functional Requirements

Functional requirements of the application are listed below:

RequirementsID	Requirement	MoSCoW	Completed?
RF1	The system shall have a location update	M	✓
RF2	The system shall allow user to call emergency contacts	M	✓
RF3	The system shall send location updates through text to trusted contacts	M	✓
RF4	The system shall record audio/video for a short period of time if panic button is pressed	M	✓
RF5	The system shall integrate Google Maps API	S	✓
RF6	The system shall implement a proximity based social media platform	S	
RF7	The system shall have a journey countdown alert	S	✓
RF8	The system shall have an alarm feature to make a loud noise to deter criminals	S	✓
RF9	The system shall implement a fake call feature	S	
RF10	The system shall have a panic button, to quickly contact emergency contacts	S	✓
RF11	The system shall implement an alternate route planner to avoid unsafe locations	S	
RF12	The system shall implement states - Standard and Alert - to improve usability	S	✓
RF13	The system shall use time filters to trigger alert mode at certain periods of time	S	

These requirements were compiled by taking into account answers to the survey I carried out, available as Appendix A. Full results are available as Appendix B. It was found that out of 74 participants, 61 of them thought a location tracker to be one of the most important features within the application. Thus, location tracking is a 'Must' in the requirements, as it is what most participants expect to be in. Another important feature, with 55 out of 76 votes, was the panic button, so this also became a 'Must'. Other features included in the requirements became 'Should' and 'Could' in terms of MoSCoW.

In terms of the requirements above that were unfortunately not implemented in this iteration of the project, these are discussed in the future plans (see section 9.3), with a strategy as to how to implement them.

3.3 Non-functional Requirements

From using my survey results, I also derived non functional requirements. These are listed below:

RequirementsID	Requirement	MoSCoW
NRF1	The system shall be reliable and robust.	M
NRF2	The system shall work with recent versions of Android OS.	M
NRF3	The system shall be intuitive and easy to use.	M
NRF4	The system shall not invade users privacy and make clear when it is tracking location.	M
NRF5	The systems features shall be fully customisable for the user.	S
NRF6	The system shall make clear what permissions it is using to the user.	S
NRF7	The system shall use as little battery power as possible while running.	S

Chapter 4

Project Design

4.1 Use Cases

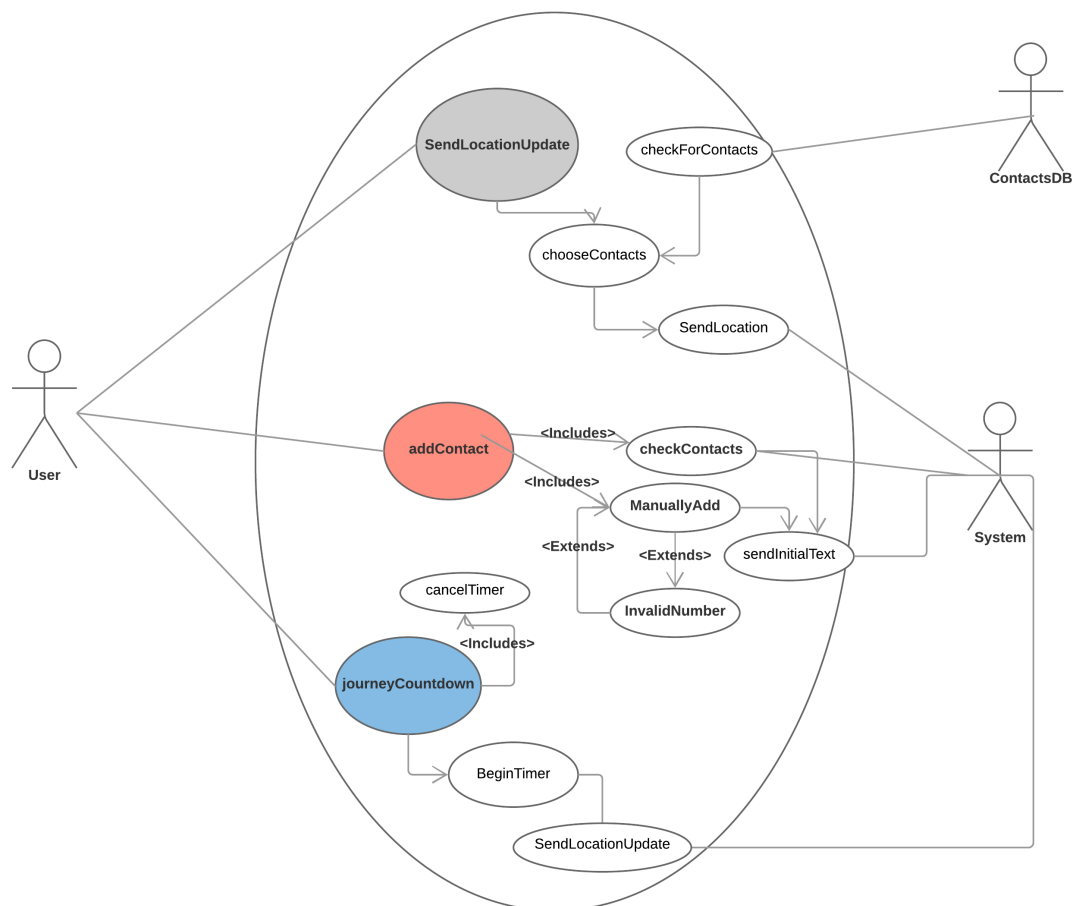


Figure 4.1: Use Case Diagram for KeepSafe

I created one use case diagram, involving three of the most important features of the application - adding an emergency contact, sending a location update, and setting a journey countdown timer. I have also added use case textual descriptions, for both features implemented and future features. These are available in appendix C.

4.2 Initial User Interface and Name

I decided before creating the application, that I would design the user interface (UI) first. This allowed me to gauge what functionality would appear on which screens. To create a UI, I used the online tool FluidUI [24], which includes an Android template, with recognisable components. This allowed me to visualise exactly what the application would look like, and would highlight if a certain area of the activity (screen) was too cluttered, or if the buttons on the navbar were too close together. Another feature of FluidUI is that it can link activities together if specified by the user. With this, I was able to check if an activity linked to another activity would work, and would suit the application. This was incredibly useful in deciding the original user interface design for the application, and gave me a good starting point to start development. During this initial stage of design I also created a name for the application - KeepSafe. I feel like this perfectly encapsulates what the application should be doing and how it works - by keeping the user safe. Figure 4.2 shows what the UI was initially intended to look like

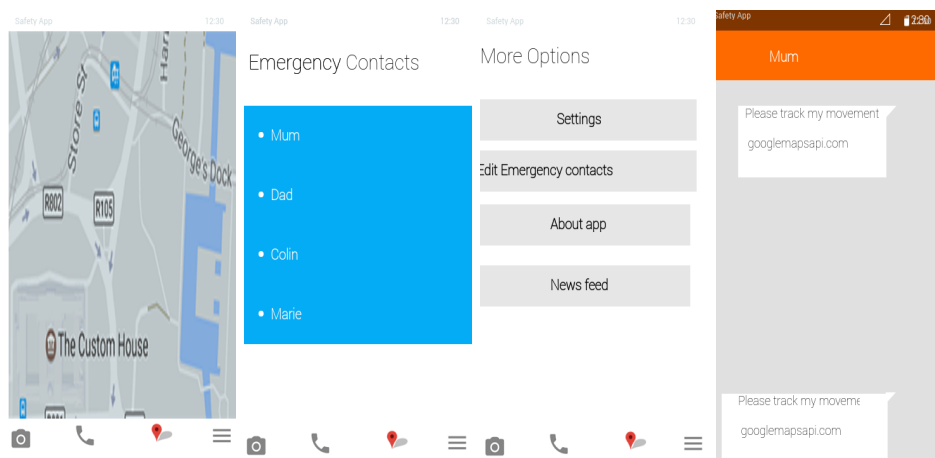


Figure 4.2: Initial UI design

Figure 4.2 shows four initial screens - a homepage, emergency contacts, a settings page, and the messaging screen (in this order). This was encapsulating all basic functionality that I aimed for.

Homepage - I intended on the application opening on the Google Maps screen, showing the users location and allowing them to view their surroundings.

Emergency Contacts - This screen would display the users stored emergency contacts, and allow them to call from within this activity. It would also allow the user to add/delete contacts as they like.

Settings - This page would allow the user to change the settings within the application, as it was to be fully customisable. It would also link to activities that were not as important to be on the navigation bar (that runs along the bottom of all screens).

Messaging - This screen would call the user's phones native messaging application, and populate the contacts with who is in the user's emergency contacts, and set the text to be a link to Google Maps with the user's latitude and longitude, allowing a pinpoint for the users location on the map.

4.3 System Architecture

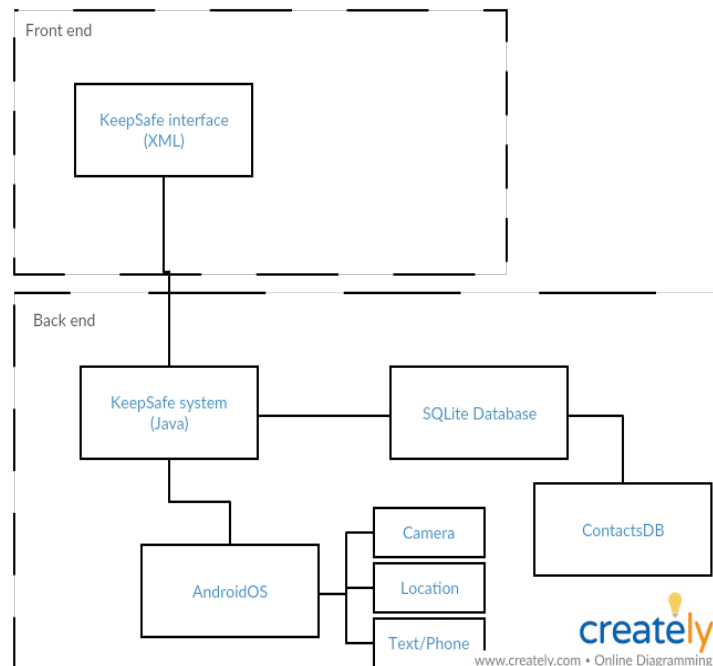


Figure 4.3: System Architecture Diagram for KeepSafe

Figure 4.3 shows a system architecture diagram depicting how KeepSafe will work. The user will interact with the front end interface of KeepSafe, which will be coded in XML, the standard layout language of Android applications. This will then link to the backend implementation of KeepSafe, which has links to the Android OS, to allow the application to access the camera to record, the phones location, and the text/phone features of the phone. KeepSafe is also linked to a SQLite Database, which is named *ContactsDB*, which stores the name and number of emergency contacts. This diagram was created using Creately [25].

Chapter 5

Implementation

Throughout this project, I implemented three prototypes, the first differing drastically from the latter two. Below I will detail each stage, and the functionality for each version.

5.1 Intended functionality

Through using the results from my initial survey (see section 3.2), I found what functionality the majority of users would like, with the most popular being the ability to send location updates (82%), a panic button (74%) and an alarm sound being played at users request (53%). With this in mind, I gradually built this functionality up over three prototypes. This allowed me to focus on getting each part working well before moving on to the next. However this was not all the application was intended to do, a significant advantage of KeepSafe is the ability to store emergency contacts within the application - meaning the user does not need to know phone numbers off the top of their head, and it makes it far easier to add contacts. I intended on importing the contacts from the phone to the application, and then allowing the user to click on them from within the application, this would link to a SQLite database and store the contacts name and number. There will also be an option to manually enter the name and number if the user prefers.

Taking inspiration from other personal safety applications - like bSafe[19] and SafetiPin[20] - I also intended on linking the camera application on the user's phone to allow for video recording/picture taking. This also doubled as having an audio recorder (as the video would record audio also). This would help reassure the user that if they were concerned about something/someone, they could have easy access to their camera to record and document what was happening. The application would allow the user to save the file to their phones internal storage and access at a time.

Another feature I intended to make available within my application is a journey countdown timer. This

would allow the user to enter a time in the application that they are allocating for their journey (for example 20 minutes). Once this timer has reached zero, the application will send out a location update to all emergency contacts stored within the application. To me, this was a very important feature, if the user is unable to access their phone, or they are in trouble during their walk, then the update will send automatically and hopefully help the user as their emergency contacts will know their location upon the countdown ending.

Lastly having a link to Google Maps API would allow users to see their location on a map from within KeepSafe. I intended on this being used more so if the user is in a location they are unfamiliar with, and may need some guidance as to how to find their way to their location. This would allow the user to quickly be taken to Google Maps to use their route planner feature. This was originally going to be the homepage to the application, however through multiple iterations this ended up being changed.

I also decided that the program I would be using to create, compile, and run the application was Android Studio [26], which is the official integrated development platform for Android. This made the whole process of developing the application simpler, as there are a lot of resources online for Android Studio, and as it is the official platform, a lot of questions surrounding it are answered online and easily available.

5.2 Back End

I created a SQLite Database, within the project. I called the database ContactsDB, and had multiple query functions within a helper class which allowed me to extract information from the database to use within the application. For example, when a contact was clicked from the users stored contacts, an alert dialog would appear, asking if the user would like to add this contact as an emergency contact. If the user selected *yes* then the insert query was called with the suitable parameters. The contact was then stored in the Contacts database which consisted of three columns - ID (the primary key), ContactName (the contacts name), and ContactNo (the contacts phone number). This was a simple solution and did not store any unnecessary data. **myDBHandler.java** was my helper class, and this contained all of the queries used to extract data to be used by KeepSafe. For example, to retrieve all the contact numbers stored within ContactsDB, this is done using the code in figure 5.1.

```
public String[] getContacts(){
    Cursor cursor = getReadableDatabase().rawQuery("SELECT contactNo FROM contacts",
        null);
    cursor.moveToFirst();
    ArrayList<String> numbers = new ArrayList<String>();
    while(!cursor.isAfterLast()) {
        names.add(cursor.getString(cursor.getColumnIndex("contactNo")));
        cursor.moveToNext();
    }
    cursor.close();
    return numbers.toArray(new String[numbers.size()]);
}
```

Figure 5.1: Querying database to retrieve contact number

By using methods such as the one in figure 5.1, I could extract the data I needed. I stored data by creating String arrays, and calling the methods in **myDBHandler.java** to populate the arrays. This is shown in figure 5.2.

```
final myDBHandler dbHandler = new myDBHandler(this, null, null, 1);

final String[] allContacts = dbHandler.getAll();
final String[] phoneno = dbHandler.getContacts();
final String[] contactname = dbHandler.getContactName();
```

Figure 5.2: Initialising the String arrays

Each class I accessed the database in, I created a new object of myDBHandler. To add contacts from stored contacts (within **showAllContacts.java**), I set an *onClickListener* on a ListView that was populated by the stored contacts. When a user selected a contact, the *onClickListener* was called and the code in figure 5.3 was executed.

```

alv.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    public void onItemClick(AdapterView<?> a, View v, final int position, long id) {
        AlertDialog.Builder adb = new AlertDialog.Builder(showAllContacts.this);
        adb.setTitle("Add contact?");
        adb.setMessage("Are you sure you want to add " + contacts.get(position) + " to emergency
            contacts?");
        final int positionToAdd = position;
        adb.setNegativeButton("Cancel", null);
        adb.setPositiveButton("Ok", new AlertDialog.OnClickListener() {
            public void onClick(DialogInterface dialog, int which) {

                String contactName1 = names.get(position);
                String contactNumber = numbers.get(position);
                ContactsDB contact =
                    new ContactsDB(contactName1, contactNumber);

                dbHandler.addProduct(contact);
                SmsManager.getDefault().sendTextMessage(contactNumber, null, "You have been added as
                    an emergency contact by " + userName + " through KeepSafe, expect some location
                    updates!", null, null);
            }
        });
    }
});

```

Figure 5.3: Adding stored contact to database

```

public void addProduct(ContactsDB contacts) {

    ContentValues values = new ContentValues();
    values.put(COLUMN_CONTACTNAME, contacts.getContactName());
    values.put(COLUMN_CONTACTSNO, contacts.getContactNo());

    SQLiteDatabase db = this.getWritableDatabase();

    db.insert(TABLE_CONTACTS, null, values);
}

public ContactsDB findProduct(String contactname, SQLiteDatabase db) {
    String query = "Select * FROM " + TABLE_CONTACTS + " WHERE " + COLUMN_CONTACTNAME + " = \""
        + contactname + "\"";

    Cursor cursor = db.rawQuery(query, null);

    ContactsDB contacts = new ContactsDB();

    if (cursor.moveToFirst()) {
        cursor.moveToFirst();
        contacts.setID(Integer.parseInt(cursor.getString(0)));
        contacts.setContactName(cursor.getString(1));
        contacts.setContactNo(cursor.getString(2));
        cursor.close();
    }
    else {
        contacts = null;
    }
    return contacts;
}

```

Figure 5.4: addProduct method to insert data into database

Figure 5.4 shows how the data is transformed from a Java array into an SQL query to insert the contact into the database.

5.3 Prototype 1

5.3.1 Implementing Prototype 1

The first prototype created for KeepSafe followed the initial design laid out in section 4.2 of this report. The application opened up displaying the Google Map API, which also showed the users current location in the form of a red marker on the screen. Along the bottom of the application was a navigation bar, similar to the one shown in Figure 4.2 of section 4.2. This linked to the native messaging application on the phone, with the text field being populated with a link to Google Maps (this approach also allows those who do not have the Google Maps application on their phone to open it on their browser). In terms of extracting the user's location using the phones GPS, I imported the LocationManager class from Android, and set up a variable that would call this class, and request the GPS longitude and latitude,

the code below in figure 5.5 is an extract of getting the users location:

```
\\RequestLocation.java

LocationManager locationManager = (LocationManager)
    getSystemService(Context.LOCATION_SERVICE);

locationManager.requestLocationUpdates(
    LocationManager.GPS_PROVIDER,
    MINIMUM_TIME_BETWEEN_UPDATES,
    MINIMUM_DISTANCE_CHANGE_FOR_UPDATES,
    new myLocationListener()
);
```

Figure 5.5: Code to extract users location

To then put this data into a format suitable to be sent to an emergency contact, I use the location-Manager variable set up in figure 5.5, to extract the longitude and latitude, and then put this into a Google Map link. In figure 5.6, this is done and added to a text message to be sent to a contact, which will be implemented in Prototype 2.

```
Location location = locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER);

if (location != null) {
String message = "A MESSAGE FROM KEEPSAFE: \n You can track " + userName + "'s location
    here: http://maps.google.com/maps?q=loc:" + String.format("%f,%f",
        location.getLatitude(), location.getLongitude());
}
```

Figure 5.6: Excerpt showing how the location update is created

Another button on the navigation bar was to a camera, which would open the users camera application on their phone and allow them to take a picture. This picture was then stored on their phone storage within their gallery. This feature was intended to be used in case the user found something/someone suspicious, they could take a photo from within the application for convenience instead of exiting and opening their camera. The last button on the navigation bar linked to settings, which was just a placeholder for future functionality.

Importantly, the first prototype was simply to ensure that the location update worked, as this is fundamentally the most important feature to the app. Based on the study I conducted in section 3.1, 82% claimed the most important feature was the location tracker/updater. This process involved linking the Google Map API to the application, I done this by linking the API to my Gmail account, and then using

```
android:name="android.permission.CAMERA"
```

Figure 5.8: Camera Permission

the key generated by Google to link to my application. From this I added permissions to the Android Manifest file (which "describes essential information about your app to the Android build tools, the Android operating system, and Google Play." [27]). The following permissions were added to ensure the application had access to the phone's current location:

```
android:name="android.permission.ACCESS_FINE_LOCATION"
android:name="android.hardware.location.gps"
```

Figure 5.7: Location Permissions

As a sidenote, the FINE.LOCATION could be changed to COARSE.LOCATION but I went with FINE as I wanted the location to be as accurate as possible. I had some issues regarding allowing the location to be read in by the application, but by changing the SDK version from 16 to 22 in the Gradle file of Android Studio. The Gradle file details to Android Studio how to compile and build the application to run on Android. I was able to run the application and see my location, and also to extract the longitude and latitude of the GPS coordinates and to concatenate them with a web URL for Google Maps.

The camera also linked with the application by using:

and allowed the user to take a picture from within the application and save it to their phones internal storage.

5.3.2 Evaluation of Prototype 1

Apart from this however no other functionality was added to prototype 1, and the application looked rather basic. After some deliberation I realised that the application did not have to open up on the Maps activity, and in fact, a lot of users may not even use this part of the application. The main crux of the project was to send location updates, and this should have been at the forefront of the development. Another drawback to Prototype 1 is the lack of emergency contacts. This feature was not developed until Prototype 2, and thus the user had no option to save contacts within the application. This resulted in the user having to type in their contacts name from within the messaging application, which was not ideal as it meant spending extra time, when it was crucial that the application worked in as few clicks

```
android:name="android.permission.READ_CONTACTS"
```

Figure 5.9: Reading Contacts Permission

as possible.

The navigation bar that was to be displayed consistently on every screen was not implemented in Prototype 1, buttons were used in a horizontal Linear Layout to emulate a navigation bar but I knew this was not going to be a permanent solution. With all of this in mind, a lot was changed between Prototype 1 and Prototype 2, which is detailed in section 7.3.

5.4 Prototype 2

5.4.1 Implementing Prototype 2

The main focus of prototype 2 was to allow the user to store emergency contacts within the application, create a panic button, and to maintain a consistent layout within the full application, using a navigation bar. In terms of the emergency contacts, I decided on accessing the users contacts stored on their phone, and store them in the application to print out. This was achieved through adding the permission shown in figure 5.9. My plan was to link this to a database, which would store the name and number of the contact selected by the user (see section 5.2).

If the user does not want to add a contact through their saved contacts, they also had the option of adding manually, which took the user to a form with name and number fields. To then send a location update to these stored contacts, I added the following code to the **RequestLocation.class**, continuing on from figure 5.6 in section 5.4.1.

```
SmsManager.getDefault().sendTextMessage(textContacts, null, message, null, null);
}
```

Figure 5.10: using the SMS manager to send the location update

Within the same method as defined in figure 5.6, figure 5.10 shows the extra line of code that calls the phones SmsManager, and sends an update. The variable *textContacts* is a String array that stores all of the numbers of emergency contacts. The user can modify if they want to send a location update to everyone in their emergency contacts, or if they want to modify who they send it to. Figure 5.10 shows the former, as this will send a text to everyone in *textContacts* immediately.

Within Prototype 2, I also created a navigation bar, created out of Radio Buttons, which allowed the user to seamlessly navigate through the application. The navigation bar was stored in a **MainActivity.java** which each other activity extended from. The advantage to this was that it meant if a new button was added, it would only be changing one class, and any new activities would only need to extend this class to also be consistent in layout, and contain a navigation bar.

A panic button was deemed an important feature to a personal safety application by 74% of the participants of the study I carried out (see section 3.1). The problem with the implementation of this was simply - what would the panic button do? It would perhaps change from user to user, and more importantly from situation to situation. Sometimes the user may favour using the panic button to call emergency services, but may also prefer it if the panic button played a blaring alarm noise to deter a criminal. To solve this, I created an option in the Settings page of the application to allow the user to pick what they wanted the panic button to do. For this implementation these were:

- Call emergency services (999)
- Record video
- Send location update
- Play alarm sound

These were implemented as radio buttons, and the user could select any option.

More functionality was added during development of Prototype 2, with the Settings page allowing the user to customise what certain buttons do. For example, the user could set the location updates to send to all stored emergency contacts without bringing up the message interface, or they could edit who the update would be sent to within the messaging interface. I also added a name variable to the application during this stage of development. This was to allow a name to appear on outgoing messages from Keep-Safe. The theory behind this choice is that not everyone is saved under recognisable names on phones, and it also makes it easier to know exactly who is messaging you their location.

All of the selected settings were stored using Shared Preferences on the phone. Shared Preferences are key-value pairs, saved in an XML file and are private to the application. This allowed me to save "panic button settings" to be one option, for example "call 999". This would overwrite the previous value attributed with "panic button settings" and would save it with "call 999" instead. A case statement within the **MainActivity.java** was used to determine what to do with each situation. This also means more options can be added in future implementations if needed. Using SharedPreferences was vital, as it allowed all settings to be saved easily and retrieved quickly. It was also not suitable to put the panic button settings in a database, as it would be complicated to extract the correct setting and apply it to a case statement.

During developing Prototype 2, I decided upon adding states to the application, starting with 2 - standard and alert (see section 5.6.2). These would be quick to change, and would result in the application changing behaviour. For example, in alert mode, a location update would be sent to everyone in the emergency contacts, without being able to be edited, regardless of the settings chosen by the user. Any future update would also be sent to everyone, with no messaging interface appearing. There is also a countdown timer in alert mode, which when the user enters a time (in seconds) the application counts down and sends a location update once the timer expires. This feature would be useful when the user is concerned about something on their walk, and decides to update their emergency contacts every 30 seconds (for example).

The interface for alert mode is different also. The top notification bar changes to blue, and there is no navigation bar. This allows the user to clearly determine which mode they are in. From within alert mode, the user can also easily phone the emergency services, or record video from within the application. As part of the usability study I conducted in semester 2, participants are asked what their opinion of this feature is, and whether any features are missing from the list of buttons in alert mode. It was noted that there was no option of an alarm sound, and a fake call feature, and those would both be implemented in future development. All of the participants did note however that this feature was useful and that they would use it.

5.4.2 Evaluation of Prototype 2

Prototype 2 was the basic application which would allow the user to store emergency contacts, send location updates, change states, use a panic button, and record video. I would take this further within Prototype 3 and focus more on usability over functionality. This prototype also contained a consistent layout among all screens, and allowed the user to seamlessly traverse the application without difficulty. A few challenges were encountered when trying to make the navigation bar work. The main challenge stemmed from Android Studio[26]. When creating a new activity within Android Studio, it prompts you for a template - if the bottom navigation bar template is selected then fragments need to be used to swap what is being displayed on the main activity, and not a new activity for each. Upon first implementing this, I realised that each time the bottom navigation bar was opening a new activity, it was opening the "Home" section of the bar, regardless of which activity. This caused a lot of issues as it resulted in a lot of code duplication, and it was very difficult to send data through the application and store it. Upon realising this, I instead implemented a horizontal radio button bar at the top of the main activity, and extended this to all other activities resulting in a quick, consistent layout. It also resulted in a lot less

code duplication and the ability to store data in a central place (either Shared Preferences or a database).

Another challenge I faced during implementing Prototype 2 was which kind of database to use. Within Android development there are multiple different types of database system, each with their own advantages and disadvantages. The type I went for was an SQLite database. From previous experience, I am very familiar with SQL-like systems, specifically MySQL, and the fact that the queries written in SQLite are very similar to those of MySQL meant I could pick them up easily and quickly. I created a helper function which held all queries for the database, and called them depending on what I needed within the application.

5.5 Prototype 3

5.5.1 Implementation of Prototype 3

As discussed in section 5.4, Prototype 2 had the majority of basic functionality and could be a standalone product. However I still had some lesser functionality to implement - such as a journey countdown displayed on the homepage, the application recognising it had not been opened before and asking the user for an initial setup, implementing a *name* variable to allow the application to be more personal, among other things. First off however, I had to ensure that the user had their location enabled on their phone. Without this the application would fail to send location updates, and other than some basic functionality, would not work correctly. To counter this, I created a function to be called each time the application is opened to check if the location is enabled. If it was not, an alert dialog would appear and explain this to the user, if the user agreed, the location would be enabled from within the application. This was a quick and easy way to ensure the application would work as intended and also meant the user did not have to leave the application to enable location.

As mentioned in section 5.4, the settings page within the application allowed the user to customise what they wanted certain functions to do, for example - change what the panic button does, or if the user wanted to send the location update to everyone instantly. If these values are not set then the application will not recognise what to do and will crash. By creating a boolean *isFirstTime*, I was able to determine if the application had run before, if it had not, then **SettingsFirstTime.java** would be run instead of **Homepage.java**. This would then allow the user to set up the application, by entering their name, to initially set up what the panic button does when clicked, and to choose if they want the location updates to send to everyone without confirmation or to bring up the messaging interface. If any of these settings are not initially set up, the user cannot start the application. There is also a reminder to add emergency contacts (however this is not forced as it is not required).

Another feature I added during developing Prototype 3 was to send an automated message to an emergency contact when added, stating that they have been added as an emergency contact by the users name, for example:

You have been added as an emergency contact by Jamie through KeepSafe, expect some location updates!

This message lets the contact know that they are chosen to be an emergency contact and that they should expect some updates. It is a simple feature but i feel like it adds a lot to the application, and it means the contact may be more likely to be checking their phone for updates.

5.5.2 Evaluation of Prototype 3

With this being the final implementation of KeepSafe for this project, I feel it was successful. The changes from Prototype 2 to Prototype 3 were more improvements to the usability of the application instead of functionality. The purpose of this prototype was to make the application more appealing to the user, and to highlight the functionality behind the application to the user in a fluid, intuitive manner. The process of this prototype was to take what had already been implemented in terms of functionality and think how to improve it. For example - adding a first time settings page means that all variables within the Shared Preference key values had a corresponding value from initial start up. This kind of addition means the application is more robust and reliable - it does not crash on startup if the user has not first visited the settings page. Within the usability study I carried out in section 8 of this report, I queried if some of these additions were helpful, and received feedback from users based on this.

5.6 Unique Features of KeepSafe

In this section of the dissertation, I will discuss some of the more unique and advanced features of KeepSafe. These are features that I feel are unique, and that set KeepSafe apart in terms of other personal safety applications.

5.6.1 Panic Button

The panic button is one of the most important components in KeepSafe. It allows the user to have an emergency button displayed on every single activity, in the same place so that the user will always know where it is. The first challenge I faced when implementing the panic button was what it would do when clicked. While I had multiple ideas on what it *could* do, I realised that I would leave it up to the user to decide what the panic button *would* do. Within the settings page of KeepSafe, there is an option

to select what the panic button should do, as figure 5.11 shows. The user can select any of the radio buttons, and the panic button will change behaviour.

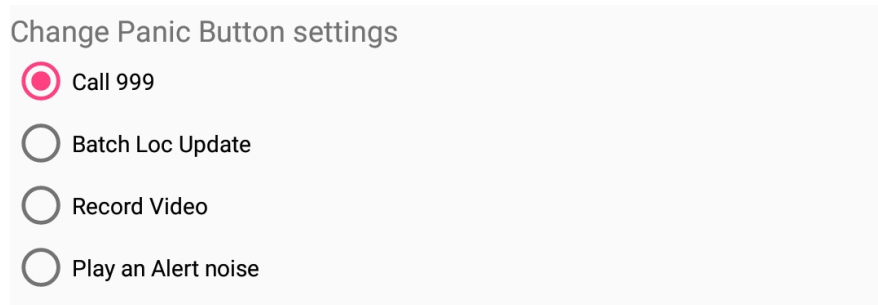


Figure 5.11: UI for choosing a panic setting

To work out which settings has been selected, I created a key value pair in the shared preferences document, with *checkPanicSettings* being the lookup value, and each setting being the execution value. First I had to assign each radio button on the activity to an *ID* so that I could check which had been checked. This was done through the XML file and can be seen in figure 5.12.

SettingsActivity.XML

```
<RadioGroup
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:gravity="start"
    android:id="@+id/panicButtonGroup">
<RadioButton android:id="@+id/Call999"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Call 999"
    android:onClick="checkPanic"/>
<RadioButton android:id="@+id/batchSend"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Batch Loc Update"
    android:onClick="checkPanic"/>
..
..
```

Figure 5.12: Defining radio buttons in XML

I define a radio group, for formatting purposes, and then define each radio button inside this group. Each radio button is given an *ID* and a method to call when it is clicked. The method *onClick* is defined in figure 5.13.

```
//Settings.java

public void checkPanic(View view) {
    // Check which radio button was clicked
    switch (view.getId()) {
        case R.id.Call999:
            this.checkPanicSettings = "phone";
            editor.putString("checkPanicSettings", "phone");
            editor.apply();
            break;

        case R.id.batchSend:
            this.checkPanicSettings = "sendLoc";
            editor.putString("checkPanicSettings", "sendLoc");
            editor.apply();
            break;
    }
}
```

Figure 5.13: Code excerpt for checking which radio button has been clicked

When a radio button is selected, *onClick* method is executed. This checks which radio button was clicked through a case statement by checking which button ID is matched. Then it puts the corresponding string into the Shared Preferences and applies (stores) it in the key value pair.

Finally, I had to specify what the application should do depending on which setting had been applied. Within **MainActivity.java**, I made a method *openPanic* which checked which values were in the pair, and executes depending on what is selected.

```
//MainActivity.java

openPanic(View view) throws IOException {

    SharedPreferences prefs = getSharedPreferences("Settings", MODE_PRIVATE);
    checkPanicSettings = prefs.getString("checkPanicSettings", "missing");

    switch (checkPanicSettings) {
        case "phone":
            Intent callIntent = new Intent(Intent.ACTION_CALL);
            callIntent.setData(Uri.parse("tel:999"));
            startActivity(callIntent);
            break;
        case "sendLoc":
            Intent in;
            in = new Intent(getBaseContext(), RequestLocation.class);
            startActivity(in);
            break;
    }

    //Continues on with other case statements
}
```

Figure 5.14: Code for checking which panic button option is selected and executing

The code snippet in figure 5.14 shows how KeepSafe works out how to execute each panic setting. This is a unique component in the application, and the other personal safety applications I tested, while

having variants of a panic button, did not allow the user to specify what they wanted, instead having a set sequence to carry out on execute.

5.6.2 Alert Mode

Another advanced feature of KeepSafe is the **Alert** mode. This allows the user to transition from the normal application (**Standard** mode) to a different, simpler interface, which is intended to make it easier for the user to interact with the application. The user can change from Standard to Alert mode either in the settings tab of the application or there is a button displayed on every activity at the bottom right. The difference between Standard and Alert is shown in figure 5.15. The difference is very noticeable, and much less cluttered, it is 4 buttons in vertical layout, and a quick countdown timer (instead of a journey countdown, detailed in section 5.6.3).

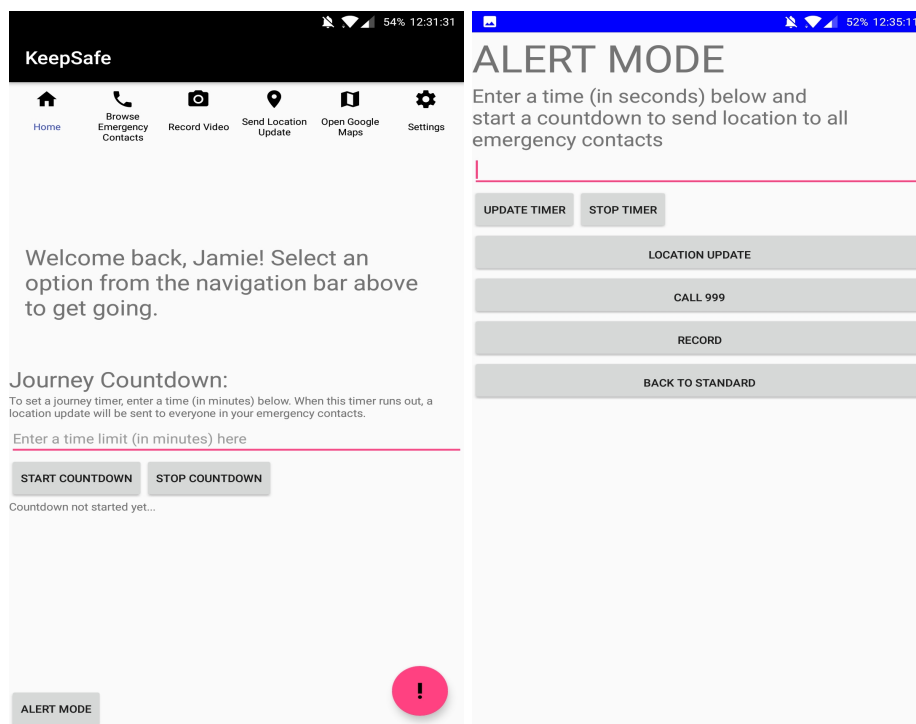


Figure 5.15: The Alert Mode Interface

Alert mode is intended to be used when the user feels they are in danger, upon changing to alert mode, the application sends out a location update to every emergency contact. There is a countdown timer, which counts down in seconds, which the user can specify how long, and four buttons:

- Send a location update
- Call 999
- Record

- Back to Standard

As evident, there is not a lot of extras on this page, only the essentials. The functionality that exists is the same however, and utilises the same classes and extensions as the Standard mode does. There is no option to edit who receives location updates in Alert mode either, with location updates being sent to everyone in emergency contacts. Within the usability study I carried out (see section 8) I asked participants if there was any features missing from this page, in the future these features would get added. For example, one participant noted that there was no alarm sound prompt on the Alert mode. Features like this would be easy to add, and would be done so in future.

5.6.3 Journey Countdown

Another feature of KeepSafe which is not available in other personal safety applications is the journey countdown timer. This is available on the homescreen of the application and allows the user to specify a time (in minutes) for their journey. If they do not cancel the timer, or navigate away from the screen, and the timer expires, a location update will be sent to all emergency contacts. This reasoning behind this feature is that users can specify how long they think their journey will take, if it exceeds this time, then something may have happened to them, and their location will be sent automatically to emergency contacts. I thought this feature was very useful, and can be applied in all situations, and would help the user feel more at ease - even if something was to happen to them, they could potentially be found through their location.

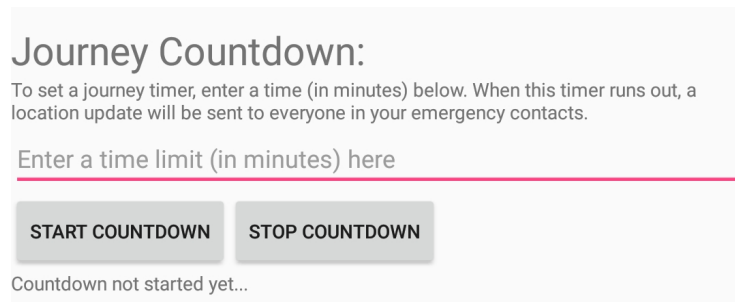


Figure 5.16: Journey Countdown Timer UI

I imported the countdown timer from AndroidOS and utilised this and its corresponding methods to set up the timer. Once the timer expired, *sendLoc* was called, getting and sending the location to contacts.

```
//Homepage.java

public void startJourneyTimer(int value){
    countdown = new CountDownTimer(value, 1000) {
        public void onTick(long millisUntilFinished) {

            String text = String.format(Locale.getDefault(), "Time Remaining %02d min: %02d
            sec",
            TimeUnit.MILLISECONDS.toMinutes(millisUntilFinished) % 60,
            TimeUnit.MILLISECONDS.toSeconds(millisUntilFinished) % 60);
            mTextField.setText(text);
        }

        public void onFinish() {
            mTextField.setText("Update sent!");
            sendLoc();
        }
    }
}
```

Figure 5.17: Setting up the countdown timer

This same method was also used for the Alert mode countdown, however it was in seconds whereas this is in minutes.

Chapter 6

Components of KeepSafe

In this section I will detail all of the activities and backend files for the project, what they are and for the activities screenshots of the layout.

6.1 Front End Components

Component Name	Description
aboutPage	A simple activity displaying information about KeepSafe, and what permissions it uses.
addContacts	A form allowing the user to manually enter a contact's name and number to be added to emergency contact.
alertPage	The Alert interface, containing a much simpler UI and implementing a quick countdown feature to send a location update in x amount of seconds.
emergContacts	The activity which shows the saved contacts within the application, and allows the user to add/delete contacts.
home	This is the initial activity, it contains a personal message to the user, and a journey countdown timer (in minutes).
mainActivity	Arguably the most important class, this is the root activity, all other activities extend this. It specifies the navigation bar and also sets the panic button's and alert button's location.
mapsActivity	Extends the GoogleMaps API and displays a map, with the users location specified with a red marker.
removeContact	Allows the user to delete a contact by entering the contact name.
requestLocation	Finds the longitude and latitude of the user and sends it to emergency contacts in whatever way the user has specified.
settings	A form with radio buttons, allowing the user to specify what they want certain features to do - panic button, and how they would like to send a location update. Also allows user to change the alert mode.
settingsFirstTime	A form that only opens on the first open from the user, and allows them to set initial settings, thus giving initial values to everything.
showAllContacts	Populates a list view with all of the users contacts stored on their phone, on click the contact gets added to the emergency contacts of the application.

Figure 6.1: Front End Components of KeepSafe

In figure 6.1 details the front end components of KeepSafe, including all of the activities used within the application. There is also a description for each activity explaining what its purpose is.

6.2 Back End Components

Component Name	Description
ContactsDB	This class specifies the database, and the columns in it. It is used to get parameters of the database.
myDBHandler	Contains functions used to create, delete, and modify the database. Also query functions to extract contact name and number. Extends SQLiteOpenHelper.
myLocationListener	Extends LocationListener, as a constructor class to set up a locationManager.

Although there is not much to the backend, in terms of the application, these are crucial. To store the contacts name and number, I set up an SQLite Database, with three columns - a unique ID, a contact name, and a contact number. I then linked this with the *emergContacts* class, allowing the user to add contacts into the database through either manually entering the name and number (through *addContact*) or through the phones already stored contacts (through *showAllContacts*). Then when the user would like to send a location update, they have the option to send it to all contacts, or edit the contacts they wish to send it to. To do this, I have created a String array, containing the number of all contacts, then I specify this array as the 'recipient' of the text message. If the contacts name is stored in the phone, the name will automatically show up, through the phones internal storage recognising it. I utilise another array to show all emergency contacts, through using a ListView in XML, and displaying a new String array that contains both name and number of each contact. This is shown below in figure 6.2

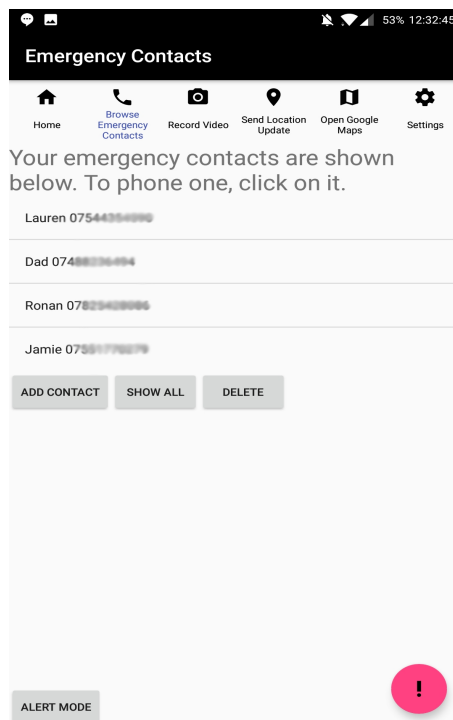


Figure 6.2: Showing users emergency contacts

Chapter 7

Testing

I carried out multiple methods of testing during the development of KeepSafe, as detailed below. This helped me keep a track of issues and fix them quickly.

7.1 Unit Testing

Unit testing is the testing of individual components or functions, to ensure they work in **isolation**. I used this method to test every function I wrote, I used print statements to test that functions were working. For example, I printed out the longitude and latitude relative to the user's current location. By doing this, it allowed me to quickly check if there were any problems with the functions I was implementing, and fix them if necessary. This kind of testing was essential to me during development.

7.2 Component Testing

Component testing is normally carried out after unit testing, to test multiple 'units' together, and ensure they work with each other. For example, I tested that on adding an emergency contact to the database, that it would also send a text to the new contact informing them of this. All of the functionality could exist, however if it does not link together correctly it can become useless and a major bug to the system. Thus, it is important to find this out as soon as possible so it can be fixed quickly.

7.3 Usability Testing

Usability testing is ensuring the application is usable. That the layout is satisfactory, and nothing is obscured or overlapping. I carried out initial usability tests, making sure the application ran well and all components of the application ran and were easily readable. After judging the user interface to be acceptable, I conducted a usability study to gather the opinion of others, who had not seen the

application before. This gave a useful insight into how usable the application was to a range of people - from students to senior citizens. It allowed for me to see if the user interface could be improved. The conclusions of the usability study are available in section 8.

Chapter 8

Usability study

8.1 Methodology

I conducted a usability study to test KeepSafe, both for functionality and for usability. I have carried out my own personal testing (see section 7) to ensure the functionality is there, and the application is robust, however my intention was that the participants would test the app thoroughly and would think of things I had not, allowing me to fix any major bugs. The application was loaded onto the test phone - a OnePlus 3 running Android 8.0.0, OxygenOS 5.0.1. More importantly though, I wanted to find out if the application had a good user interface, and all users would be able to easily navigate the application. As I was familiar with the application (having created it), it was important for me to get other opinions on the interface, and ensure it was successful.

For this study, I aimed to have around 7-10 participants, recruited through word of mouth. I gathered participants from a range of backgrounds, and age ranges as it was important to get a broad opinion of the application as it is marketed towards everyone, and should be accessible to everyone. There were students, retired senior citizens, and a range of service jobs included in the survey. The age range was also varied, all of the age ranges being represented in the study (available in Appendix E). I planned on conducting a semi-structured interview with each participant, in which I would ask them a series of questions and tasks to complete within the application. These questions/tasks are evaluated in section 8.2, but there is a brief description below in figure 8.1.

Question	Description
What age range are you in?	This was to gauge which age ranges responded well to the application and how accessible the application was.
Which gender do you identify with?	This was to check which genders responded well to the application, and relate to section 2.1.
Complete the settings form.	This was the first task, and asked the participant to complete the initial settings form to set initial values for the application. They were then asked what their opinion of the interface was.
Add an emergency contact.	This task tested one of the most important features in KeepSafe, the ability to store contacts within the application. It was important that this task was easy to complete and robust.
Trigger a panic alarm.	This task asks the participant to click on the panic button. This was to test whether the panic button was clear and the participant would know how to trigger a panic.
Change the panic button settings.	This required the participant to navigate to settings, and change the panic settings. This was to test whether it was clear that the settings was where this would be done. The participant would then be asked if there were any features missing.
Send a location update.	This was the most important feature of KeepSafe, so I wanted to test how simple this task was and ensure it was accessible to all participants.
Phone an emergency contact.	This asked the participant to navigate to the emergency contacts page and click on a contact. This was to test whether this was a noticeable feature and also test whether participants would read the help text on each page (as it was stated how this would be done on the page).
Activate Alert mode within KeepSafe.	One of the unique and advanced features of KeepSafe, I wanted to ensure this feature was easy to navigate to and users could notice what mode they were in - standard or alert.
Set a journey countdown timer.	Another advanced feature of KeepSafe, each participant would be tasked with setting a countdown timer and cancelling it. This was to ensure this feature was noticeable to the user and the functionality behind it worked.
Would this application be useful?	Gauging the participants opinion of the application and if they would this.

Figure 8.1: Questions in usability study

Before conducting the interview, each participant was asked to sign a consent form. This was a fairly standard document, stating that they were free to leave the study at any time, and that KeepSafe would be uninstalled completely from the test phone after their interview. This is available in Appendix D.

8.2 Evaluating Usability Study

8.2.1 Initial Thoughts of Application

As noted in the section 5.5 regarding Prototype 3, on startup KeepSafe prompts the user to set initial values for the settings. I asked the participants what they think of this feature, and what they think about the layout of this activity. 6/7 responded with positive notes about this feature, citing that it was useful to let the user customise the application straight away. However 66% of participants had an issue with understanding what some of the options meant. For example, they did not understand the different location update settings, and enquired for more information. One participant offered a solution to this, asking for a question mark button that showed a message popup to appear if the user clicked on it. This popup would be a useful addition to the application, and I can apply this logic to any section of the application which may be ambiguous or confusing to the user.

8.2.2 Adding a Contact

Participants were asked to add a new emergency contact to the application using their personal phone number. If they were uncomfortable with adding their personal data to the application, I offered my phone number instead, as the user would still be able to test the application this way. This question was designed to gauge whether the process behind this was simple - as it is one of the most important functions of the application. As the application will open on the homescreen, it was made clear if the user could understand the navigation bar and what each button on it linked to. All 7 participants succeeded in adding a contact (2 preferred to use the test phone number) and there were no issues on this front. As the user was typing in a number manually, it was only this interface they were using - not selecting from imported contacts. Each participant understood the difference between manual and from phone contacts and all selected manual. The results from this question helped in terms of gauging usability as all participants were able to add an emergency contact with no assistance.

8.2.3 Triggering Panic Alarm

On every screen of KeepSafe there is a panic button, however I wanted to test if it was clear what the functionality was behind the button, or if the participant would know to click the button. Through the

initial set up of the application the participant had already selected what the panic button was to do on click. The alarm button, shown in figure 8.2 is always situated on the bottom right of each screen.



Figure 8.2: Panic Button

As evident from figure 8.2, the user could easily look over this and not click it, or even know it is there. I wanted to test how true this was. Out of the 7 participants of the study, 5 clicked the button without hesitation, knowing that it was the panic button. However 2 were unsure and asked for assistance. I pointed out that the button was on each screen and not to worry about clicking something if they wanted to test it. I found that it was two of the older participants that this applied to, they were unsure about the button and so did not click it (see section 8.2.10). One participant suggested making the button larger on screen, while another suggested changing the exclamation point to the word "Panic" to be more obvious. This feedback was useful, and I would implement both suggestions in the future.

8.2.4 Configuring Panic Button Settings

Within the settings page of KeepSafe, the participant was asked to change what the panic button does. There are four options (see section 5.6.1) to choose from. This was to test whether it was clear to go into the settings page of KeepSafe first, and then find how to change the panic button settings. All participants realised they needed to go to the settings page, and 6/7 changed the settings without any assistance. One participant took slightly longer to change the settings, and had to ask what some of the options meant (for example "Batch Loc Update"). Using the suggestions from section 8.2.1, I could implement a helper dialog box to explain these terms in a future version of the application. However the radio buttons made sense to every participant and the majority knew how to change the settings. In terms of features that could be added to the panic button, one participant suggested implementing a fake call feature, in that the phone would ring (similar to the alarm sound) with the users specified ringtone, and the user could pretend to answer the phone. This feature is definitely something I would consider implementing in a future version, and was a *should* in my requirements (see section 3) but I was not able to implement it.

8.2.5 Sending a Location Update

Sending a location update is the fundamental function of KeepSafe, so it was very important for it to be as simple as possible. Participants were asked to send a location update to emergency contacts. Depending on the initial setting they had selected for sending location updates, this was either going to send the message to everyone in contacts, or display the phones native messaging application with the contacts in the *TO* section of the message. While every participant was able to send the update, some were confused with the outcome. 3 of the participants were unsure if the update had sent as the application did not bring up the messaging interface (and they had not realised they had selected to batch send). The same 3 also commented on the fact they did not like that the application send it without warning them first. I explained that this was due to the setting they selected but it may be useful to also have a popup on screen saying to the user that their update has been sent. Currently, after clicking the location update section of the navigation bar, the screen in figure 8.3 is shown.

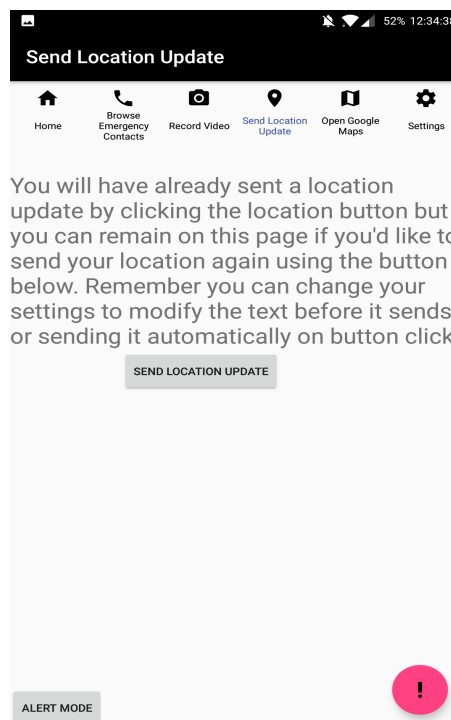


Figure 8.3: Location Update screen

This is quite wordy but does state that a location update had been sent. 3 participants suggested making the location update screen less wordy, as it was not very clear to read. It was generally the older participants who were unsure of this, which is discussed more in section 8.2.10. Overall however, it seemed that all participants recognised what to click to send an update, and all received a text (either through the test phone or their phone) proving the location worked.

8.2.6 Phoning emergency contacts

Phoning an emergency contact from within the application was a *must* within my requirements (see section 3). I wanted to gauge whether it was clear to first navigate to *Browse Emergency Contacts* and then click on the contact.

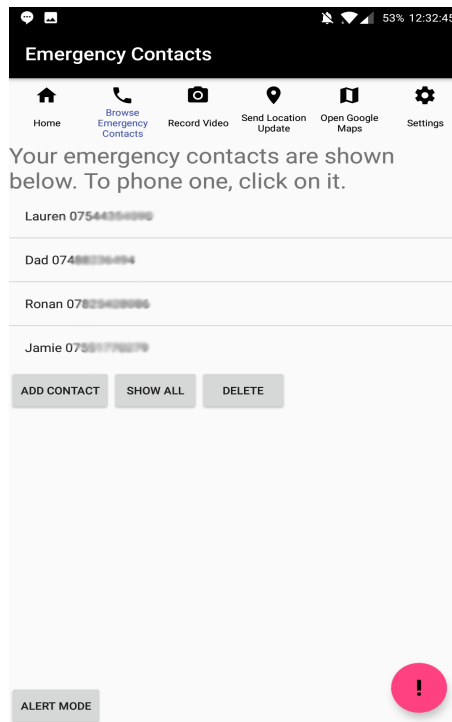


Figure 8.4: Browse Emergency Contacts

As seen in figure 8.4, there is a text box which explains how to phone an emergency contact (by clicking on it). 6/7 participants were able to phone without hesitation, one participant was looking for a button beside each contact, and suggested this afterwards. I wanted to see if participants would read the text on each screen, or just skip over it. By carrying out this task I found that while the majority do read it, some do not and that means the application has to be more intuitive and not really on text to help the users.

8.2.7 Activating Alert Mode

An advanced feature of KeepSafe is the different states - Standard and Alert. I wanted to test whether this is noticable and that users would know the difference. I asked participants to open the Alert mode of the application, and tested whether they were able to easily change. All of the participants thought this feature was useful, and could tell they were in Alert mode. 5 participants stated they liked the simplicity of interface within Alert mode, and would use the feature if they felt in danger. In terms of features missed within Alert mode, 2 participants noted that there was no alarm sound feature, but every other

panic button setting was. In future versions of the application, this will be rectified and added. All in all, Alert mode was seen favourably by the participants, and no participant had difficulty in navigating between the two states. One participant suggested implementing a loop feature on the countdown feature within Alert mode (which counts down in seconds, see section 5.6.2) so that a location update will be sent every x amount of seconds, which the user specifies. This would be implemented in future versions as it is a very good addition to Alert mode.

8.2.8 Journey Countdown

Participants were asked to set a journey countdown on the homepage. All participants were able to complete this task quickly with no questions or assistance needed. The participants liked the idea of a journey countdown, with one stating it is their favourite feature of the application. All participants said they would find this feature useful, and are glad it is on the homepage. A participant discovered that the countdown can only be done in minutes, there is no option for seconds. This is due to the `EditText` field in XML using a numeric keyboard layout to make it easier for the user only type in numbers. It also means there will be no exceptions as it can only have numbers, no letters.

8.2.9 Final Thoughts on Application

The final question of the usability study asked if the participants thought the application would be useful. The majority of participants agreed the application is useful and that they would use KeepSafe themselves (5/7 participants). 2 participants admitted they would not use the application but this is due to them not having a smartphone, and not using their phone that often. It is worth noting these two participants were older, and are not as confident with technology as younger participants were.

8.2.10 Age Difference

Throughout the study, it was clear that the older generation (46 and over) were significantly less confident in using the application than those younger than 46. I recruited two older participants as I knew they would contribute a lot to the study and offer a different insight into KeepSafe. A lot of the confusion surrounding the application was from the two older participants, but as the application is intended for anyone, this was important to discover, and it will allow the application to be improved more for the older generation. From superficial aspects of the application, such as the buttons being too small, or font being unreadable, to functionality not being sufficiently explained, it was vital to get a range of participants to find out what was not working within the application.

8.2.11 Conclusion of Study

Overall, the study was successful, and it proved that, although there are some issues with the application, it was a success. The general consensus was that the application was well laid out and the majority tasks laid out in the study were simple to complete, however there were some exceptions.

The first two questions of the study were asking the participants age range, and their gender. Out of the 7 participants who took part, there were 4 females, and 3 males, and I managed to get a mix of age ranges, this helped me gain insight into ensuring the application is accessible to all, not just younger people. I was glad to get more females to do the study, as this application, as referenced in section 2.1, is more geared towards females than males.

Chapter 9

Conclusion

9.1 Achievements

Discussed in the introduction to this report (section 1) were the main aims and objectives of this project. To reiterate, the main aim of this project was to design and implement a personal safety application that would improve the safety of the users. While it has been hard to identify whether the safety of the users has improved or not, based on the initial survey results (see section 3.1) and the usability study (see section 8), this aim has been met. Through both surveys, it was clear that participants thought that a personal safety application would make them feel safer. In the initial survey carried out, 97% of participants feel like a personal safety application could make them feel safer, and 71% of usability study participants claimed they would use the application to improve their personal safety.

In terms of achievements within the application, all major functionality was implemented (including location updates, emergency contacts, panic button, etc.) along with some extra functionality to make KeepSafe stand out among other personal safety applications. This includes features such as the state changes, journey countdown, and recording video/audio. Anything that was not able to be implemented is mentioned in section 9.3, with an explanation on how it could be implemented.

Before developing KeepSafe I had only created two mobile applications, one through Java and the other through AppInventor [28], an online block tool for programming. With this in mind, creating KeepSafe was a constant challenge in terms of development and researching.

9.2 Limitations

Despite achieving the aim of this project, there are certain features and limitations to KeepSafe. For example, some functionality laid out in section 3.2 was not achieved during this project (although is discussed in future plans, see section 9.3). With time constraints and other features taking longer to

implement than first thought, the most important features were prioritised.

Another limitation of this application was that while it displays well on the test phone - OnePlus 3 running Android 8.0.0, OxygenOS 5.0.1, it does not adapt well onto other phones, however as it was being run and tested on the same phone, it was not a priority to fix this. To overcome this, layout constraints could be added to the XML files for each activity to bound them to a set position.

9.3 Future Plans

Despite the application implementing the majority of key features that were requested from the initial survey, and from my experience with other applications, there were still some that I was not able to implement.

9.3.1 Time Filters

If I were to continue developing KeepSafe, I would implement some form of time filter - for example, if the user accesses the application from a certain time. Based on my initial survey (found in Appendix B) I found between 6pm and 6am was the most common time to feel unsafe, with 77% of participants agreeing to this. Using this, I would implement a time filter, in which if the user opens up the app during these 12 hours (or hours set specifically by the user) then the application would open in the **Alert Mode**, as this is the quicker of the two interfaces to interact with. The time filter range would be customisable, as is the rest of the application, allowing the user to specify what time range they would like this to apply to, they can also disable the feature if need be. This feature is not available on the two personal safety applications I tested, bSafe[19] and SafetiPin, [20] and I feel as though it would add a unique selling point to KeepSafe. Unfortunately, I did not have the time to implement this feature, as I encountered challenges during the applications development that caused my schedule to be delayed, these are described in evaluation sections of each prototype (section 5.3.2, 5.4.2, 5.5.2).

9.3.2 Proximity Based Social-Media

During the initial study I undertook to gather requirements and people's reaction to a personal safety application, I offered an option to implement a proximity based social media platform. This would allow users of KeepSafe to let other users know what was happening around them, even if they did not have direct contact with them. My initial thoughts on this would be, KeepSafe would display messages uploaded to an SQLite database, containing the author of the post and the message, along with the location at which the message was posted (using the same longitude and latitude extraction function as the main application). I would then implement a display function that, if the user was within a certain range of the post, it would display. This would be done by creating a radius around the longitude and

latitude of where the message was posted, and then only displaying the message to other users when they are within that radius. The idea behind this is that users can help each other, and report things that they may not deem noteworthy enough to tell the emergency services. For example, if they notice a certain person always hanging out while they are walking, they could post it on this platform to let others know to watch out for them. This would be another unique point, as the two application I tested, bSafe and SafetiPin, did not have this feature or anything related to it.

9.3.3 Fake Call Feature

With the initial study, I found that 40% of the participants would welcome a fake call feature. This would be triggered through the panic button, and would be an option to select in the settings. On triggering the panic button, KeepSafe would take the user to an activity displaying an incoming call interface, and would play a ringtone to emulate a phone call. KeepSafe would not extend the call feature from the phone, but would create a default interface for the user, such as the one in figure 9.1.

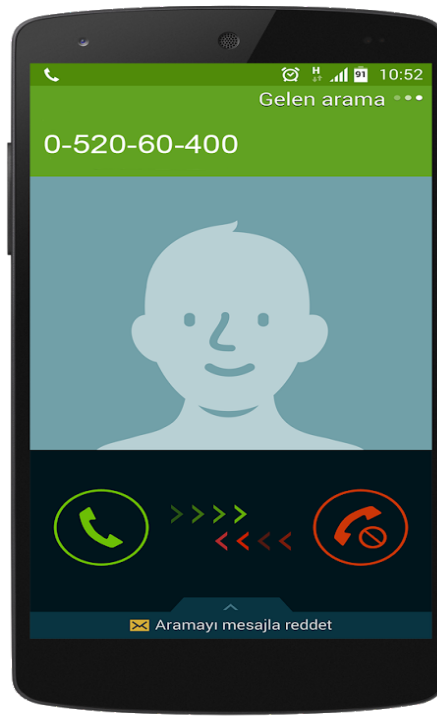


Figure 9.1: Example fake call interface

This example is taken from a fake call application available on the Google Play store called "Fake Call 2" [29]. While there are applications available for this, I feel it would be useful to include this in the personal safety application instead of having two separate applications.

9.3.4 Alternate Route Planner

In the same way Google Maps will reroute users if the original route is busy, or will take longer than an alternate route, I plan to implement an alternate route planner for KeepSafe. The theory of this would be that users could score areas out of 10 in terms of safety, KeepSafe would store the longitude and latitude of this and extend it by a 500m radius. This would work in conjunction with the social media platform, explained in section 9.3.2. If the users longitude and latitude matched with a heavily scored area, then an alert would display and inform the user, they could then work around this and walk a different way. The scores would be averaged from all users and displayed to each user. This would be a very interesting feature to include, as my research with other safety applications, bSafe and SafetiPin, neither had this feature. I feel like it would add another dimension to KeepSafe and would improve the users safety as they would know to avoid certain areas and can help other users by scoring areas they felt unsafe.

References

- [1] Orla Thérèse McCarthy, Brian Caulfield, and Margaret O'Mahony. How transport users perceive personal safety apps. *Transportation research part F: traffic psychology and behaviour*, 43:166–182, 2016.
- [2] Delphine Christin, Christian Roßkopf, and Matthias Hollick. usafe: A privacy-aware and participative mobile application for citizen safety in urban environments. *Pervasive and Mobile Computing*, 9(5):695–707, 2013.
- [3] KL Uthpala Senarathne Tennakoon and Daphne G Taras. The relationship between cell phone use and sense of security: A two-nation study. *Security Journal*, 25(4):291–308, 2012.
- [4] Jennifer L. Truman and Michael R. Rand. Criminal victimization, 2009, 2009. Available from <https://www.bjs.gov/content/pub/pdf/cv09.pdf>. Accessed 29/09/2017.
- [5] Deborah Kirby Forgays, Ira Hyman, and Jessie Schreiber. Texting everywhere for everything: Gender and age differences in cell phone etiquette and use. *Computers in Human Behavior*, 31:314–321, 2014.
- [6] Tony Charlton, Charlotte Panting, and Andrew Hannan. Mobile telephone ownership and usage among 10-and 11-year-olds: Participation and exclusion. *Emotional and Behavioural Difficulties*, 7(3):152–163, 2002.
- [7] Deloitte Global Mobile Consumer Survey. Device penetration among uk adults 2012-2017. 2017. Available from [DeloitteGlobalMobileConsumerSurvey](#). Accessed 25/09/2017.
- [8] Huizhong Wu. Indian supreme court upholds death sentences in delhi gang rape, 2017. Available from <http://edition.cnn.com/2017/05/05/asia/india-gang-rape-death-penalty/index.html>. Accessed on 30/10/2017.
- [9] BBC News. India 'gang-rape': Student, friend attacked on delhi bus, 2012. Available from <http://www.bbc.co.uk/news/world-asia-india-20753075>. Accessed on 30/10/2017.

- [10] Huizhong Wu. India says every phone must have a 'panic button' by 2017, 2016. Available from <http://money.cnn.com/2016/04/27/technology/india-smartphone-panic-button-rape/index.html> Accessed 30/10/2017.
- [11] Olivia Solon. Snapchat's new map feature raises fears of stalking and bullying, 2017. Available at <https://www.theguardian.com/technology/2017/jun/23/snapchat-maps-privacy-safety-concerns>. Accessed 29/09/2017.
- [12] Nishi Tikku, Neha Menon, Dhanashri Phatak, and Venkatesh Vaasudhevan. On the spot: An android application for personal safety, location based alarms and location based reminders. *International Journal of Computer Applications*, 171(8):41–45, Aug 2017.
- [13] Justin W. Patchin. 2015 cyberbullying data. Statistics available at <https://cyberbullying.org/2015-data>, accessed 18/10/2017.
- [14] Kumiko Aoki and Edward J Downes. An analysis of young people's use of and attitudes toward cell phones. *Telematics and Informatics*, 20(4):349–364, 2003.
- [15] Jack L. Nasar, Bonnie Fisher, and Margaret Grannis. Proximate physical cues to fear of crime. *Landscape and Urban Planning*, 26(1):161 – 178, 1993. Special Issue Urban Design Research.
- [16] Filippo Dal Fiore, Patricia L. Mokhtarian, Ilan Salomon, and Matan E. Singer. “nomads at last”? a set of perspectives on how mobile technology may affect travel. *Journal of Transport Geography*, 41(Supplement C):97 – 106, 2014.
- [17] JACK NASAR, PETER HECHT, and RICHARD WENER. ‘call if you have trouble’: Mobile phones and safety among college students. *International Journal of Urban and Regional Research*, 31(4):863–873, 2007.
- [18] StaySafe. Personal safety app. Available at <https://www.staysafeapp.com/>, accessed 18/10/2017.
- [19] bSafe. "Never Walk Alone" personal safety app. Available at <https://getbsafe.com/>, accessed 18/10/2017.
- [20] SafetiPin. "Supporting Safer Cities" Personal Safety app. Available at <http://safetipin.com/>, accessed 18/10/2017.
- [21] Google. Google keep. Available at <https://www.google.com/keep/>, accessed 18/10/2017.
- [22] Mobile operating system market share worldwide, 2018. Available from <http://gs.statcounter.com/os-market-share/mobile/worldwide> Accessed 15/03/2018.
- [23] GitHub. Leading software development platform. Available from <https://github.com>. Accessed on 17/04/2018.

- [24] Fluidui - create web and mobile platforms in minutes, 2018. Available from <https://www.fluidui.com> Accessed 15/03/2018.
- [25] Cinergix Pty. Ltd. Creately, 2018. Available from <https://creately.com>. Accessed on 28/03/2018.
- [26] JetBrains Google. Android studio. Available at <https://developer.android.com/studio/index.html>, accessed 19/03/2018.
- [27] App manifest overview. Available from <https://developer.android.com/guide/topics/manifest/manifest-intro.html> Accessed 17/03/2018.
- [28] Massachusetts Institute of Technology. App inventor, 2012. Available from <http://appinventor.mit.edu/explore/>. Accessed on 27/03/2018.
- [29] Mango Yellow Studio. Fake call 2. Available from <https://play.google.com/store/apps/details?id=com.anydroid.fakecall2&hl=en>. Accessed on 15/04/2018.

Appendix A

Survey Questions

1. What is your age, in years?
2. What gender do you identify with?
 - Male
 - Female
 - Rather not say
3. Have you ever felt unsafe while walking alone outside?
 - Yes
 - No
4. Is there a particular time of day that you might feel less safe while walking alone outside?
 - 12am-6am
 - 6am-12pm
 - 12pm-6pm
 - 6pm-12am
 - None, I don't think I feel unsafe at any particular time
5. Which of the following situations would make you feel unsafe while walking alone outside?
 - Walking through a busy shopping centre
 - Walking past a pub at midnight
 - Walking through an unlit residential area - Hearing loud shouting behind you
 - Walking to the supermarket during the day
 - Walking without your phone
 - None, I don't think I feel unsafe in any particular situation
 - Other

6. Have you ever used a mobile phone to make you feel safer while walking alone outside?

- Yes

- No

7. Have you ever used a 'personal safety application' to make you feel safer while walking outside alone?

-No

-Yes (please specify what app below)

8. In your opinion, what is the 3 most important features for a personal safety app?

(If you wouldn't find a personal safety application useful, you only need to select the one choice.)

- Location tracker

- Alarm sound (plays loud alarm when activated)

- Panic button (calls emergency contacts instantly)

- Alternate route planner to avoid unsafe areas

- Selfie mode

- Fake incoming call button

- None, I don't think I would find a 'personal safety app' useful

- Other

9. Would you be comfortable with your location being tracked as part of a 'personal safety application'?

- Yes, all the time

- Yes, as long as I could turn it off

- Not at all

Appendix B

Survey Results

1.

Age Range	Count
16-25	35
26-36	5
36-45	5
46-55	11
56+	18

2.

Gender	Count
Male	26
Female	48

3.

Felt Unsafe?	Count
Yes	61
No	13

4.

Unsafe Time?	Count
12am-6am	27
6am-12pm	3
12pm-6pm	5
6pm-12am	30
None, I don't think I feel unsafe at any particular time	9

5.	Unsafe Situation?	Count
	Walking through a busy shopping centre	1
	Walking past a pub at midnight	29
	Walking through an unlit residential area	52
	Hearing loud shouting behind you	50
	Walking to the supermarket during the day	0
	Walking without your phone	20
	None, I don't think I feel unsafe in any particular situation	4
	Other	3

6.	Mobile phone to make you feel safer?	Count
	Yes	52
	No	22

7.	Mobile app to make you feel safer?	Count
	Yes	2*
	No	72

Both answers are disregarded as misunderstood question and answer does not apply.

8.	Most important feature (pick 3)	Count
	Location Tracker/Updates	61
	Panic Button	55
	Alarm Sound	39
	Fake Incoming Call	30
	Alternate Route Planner	15
	None	3
	Selfie Mode	0

9.	Comfortable with location tracking?	Count
	Yes, all the time	15
	Yes, if I could turn it off	55
	Not at all	4

Appendix C

Use Cases

C.1 SendLocation

Name: sendLocation

Description: Send users location coordinates to an emergency contact

Actors User, Contact

Preconditions:

- Phone location is enabled
- Contact is stored in application as an emergency contact
- Map button on UI is clicked on

Basic Flow:

- System derives location from gps
- User selects a contact to send it to
- System sends text message with coordinates to contact

Alternate Flows:

- User cancels the request
- Application returns to homescreen

2.

- Contact is not stored in app
- User adds contact

- System derives location from gps
- User selects a contact to send it to
- System sends text message with coordinates to contact

Exceptional Flows:

1.

- Location is not enabled
- Prompt user to turn on location permissions

2.

- User has no signal on phone
- Message sends when user gets signal again

Postconditions:

- User has sent a location update through the application

C.2 addContact

Name: addContact

Description: Add an emergency contact to the app

Actors: User, System

Preconditions:

- User selects the add emergency contacts button

Basic Flow:

- User enters name for the contact
- User enters a valid (11 digit) number
- System accepts contact and adds to emergency contact list

Alternate Flows:

- User selects 'import contact'
- The system accesses the phones contacts
- User selects contact they want to add

Exceptional Flows:

- User does not enter a valid number for contact
- System prompts user to fix this or return to home screen

Postconditions:

- The system has added contact to the emergency contact list

C.3 firstOpen

Name: firstOpen

Description: What happens when the application is opened for the first time

Actors: User, System

Preconditions:

- User has not opened up the application before

Basic Flow:

- The system will prompt user to pick privacy options
- The system will prompt user to select what screen they would like to open on
- The system shall ask user to customise the bottom screen
- The system shall give a short tutorial on how to use application

Alternate Flows:**Exceptional Flows:****Postconditions:**

- The user has successfully customised the app to users liking

C.4 setJourneyCountdown

Name: setJourneyCountdown

Description: User sets a countdown, once expired the system will send a location update

Actors: User, System

Preconditions:**Basic Flow:**

- The user shall specify a time to countdown from (in minutes)

- The system shall begin a countdown
- Once the timer has expired, the system shall send a location update to all emergency contacts

Alternate Flows:

- The user cancels the countdown during it

Exceptional Flows:

- The user enters an invalid time
- The system notifies the user

Postconditions:

- A journey countdown has been set

C.5 PanicButton

Name: PanicButton

Description: Activation of panic button

Actors: User, System

Preconditions:

- Panic Button is selected

Basic Flow:

- The system will check to see what panic button option is selected in settings
- The system will carry out the appropriate response

Alternate Flows:

Exceptional Flows:

Postconditions:

- The panic button will be executed

C.6 AccessingSocialMedia

Name: accessingSocialMedia

Description: Accessing the social media portion of the app

Actors: User, System, messageDB

Preconditions:

- User has location turned on
- User selects social media option in app

Basic Flow:

- The system will set up a proximity and collect messages that were given within this proximity based on their location in the messages database
- The system will display these messages

Alternate Flows:

Exceptional Flows:

- The user's location is unknown
- The system will alert user and prompt them to turn on location

Postconditions:

- Social media segment will be running

C.7 PostComment

Name: PostComment

Description: Post a comment on a social media segment

Actors: User, System, MessageDB

Preconditions:

- Location is turned on
- Social media segment is open

Basic Flow:

- The system shall allow user to compose a message
- The user shall input their message
- The system shall save their message into a database, with the message, the author, and the location it was posted at

Alternate Flows:

1.

- The system shall allow user to compose message

- The user submits an empty message
- The system will prompt user to enter characters in message

2.

- The system shall allow user to compose message
- The user quits the app halfway through
- The system saves the message to drafts
- User can go in and find message in drafts and upload it

Exceptional Flows:

Postconditions:

- The user will have successfully added a message to the media board

Appendix D

Consent Form

Usability study for KeepSafe

Thank you for choosing to take part in my usability study for KeepSafe. KeepSafe is a personal safety application that allows its users to send location updates to selected contacts, and also allows them to record video, and call contacts.

For this study you will be using another phone, however you will be asked to enter your phone number and have your phone at hand, so that you can test the location update functionality. If you object to this, you can use the test phone number, this is provided in the questionnaire. I will also ask you to complete a few tasks in the app, like adding contacts, changing state, etc.

None of the data added to the app will be stored after you have finished the study, as the app will be uninstalled.

You must be 18 or over to take part in this study and you are free to leave at any time.

Participant Name:

Participant Signature:

Date:

Appendix E

Usability Questions

KeepSafe Usability Study

1. What age are you? Please circle the correct range.

18-25 26-35 36-45 46+

2. What gender do you identify with? Please circle the appropriate answer.

Male Female Rather not say

3. Complete the settings form. What do you think of this interface?

4. Add a number to the emergency contacts list. (If you would rather not enter your personal number add this XXXXXXXXXXXX). Do you like that it sends a text to the new contact?

5. Use the app to trigger a panic alarm. Can you think of any way to improve this?

6. Change the panic button settings to something else. Are there any you think are missing?

7. Use the app to send a location update to your emergency contacts. Is this process suitable?

8. Phone an emergency contact from within the app.

9. Activate the **ALERT** mode from within the app. What do you think of this feature? Is there

anything missing?

10. Set a journey countdown from within the application. Do you find this feature useful? (Please cancel the countdown after this)

11. Do you think you would find this app useful?