Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Petter Flytøren

Manual monitoring of grazing sheep

Documentation and testing of software developed for inspection tours of grazing sheep

Master's thesis in Informatics
Supervisor: Svein-Olaf Hvasshovd
June 2020

NTNU
Norwegian University of Science and Technology

Petter Flytøren

Manual monitoring of grazing sheep

Documentation and testing of software developed for inspection tours of grazing sheep

Master's thesis in Informatics
Supervisor: Svein-Olaf Hvasshovd
June 2020

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Kunnskap for en bedre verden

Acknowledgements

I would like to thank my councillor Svein-Olaf Hvasshovd, Professor at the institute of Computer Science at NTNU Trondheim, for continued guidance and input on how the application should be designed and how it should function in order to be the best that it can be. He has a lot of experience with the subject and so his input has been invaluable to the development of the application.

I would also like to thank Lars Emil Knudsen, Assistant Professor at the institute of Computer Science at Ostfold University College, which has helped a lot with Android programming, both in suggesting what services I could implement to cover the needs of the application, and also with any questions I had about specific programming issues that I encountered.

Lastly I would like to thank my focus group for helping me test the application and provide valuable feedback.

Abstract

Each year when the Norwegian sheep farmers let the sheep out to go grazing, the farmers have to go on weekly inspection tours to look after the sheep. During these inspection tours, the farmer needs to record where he/she has travelled, observations of sheep (how many and where), and also any other potentially relevant observations. Currently this is done by hand with pen and paper, and so for this thesis the student shall develop an Android application to allow farmers to do this digitally and partly automatically. The application shall work both online and offline, and so it is important that it has the ability to download maps in advance and that the data storage system can hold data until the user is online again. The finished application shall be tested using a focus group based method, which includes a usability test, a user experience test and an interview. From this plus the author's and councillor's own testing, a conclusion will be drawn about the current state of the application, what could've been done differently, and what a future solution could/should include.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 About the author

Petter Flytøren is an informatics student at NTNU Trondheim with this being his master thesis. Previously he studied a Bachelor in informatics at Ostfold University College. He has previous experience with developing mobile applications, specifically in Android, from the subject ”ITF21013 Android Programming” at Ostfold University College. In addition to this he was also a teacher’s assistant in that subject the year after he had taken it.

1.2 The assignment

Each year the Norwegian sheep farmers let their sheep out to go grazing in the warmer months of the year. During this time the sheep farmers have to do manual inspection tours to track the location and state of the sheep. They also need to record the route of their inspection tour. This is currently all done manually with pen and paper.

As current mobile technologies are quite sophisticated and also quite accurate, a lot of this can be automated and/or made considerably faster. So for this thesis, an application will be developed and documented.

1.3 Contractor/Councillor

The client for this project is Svein-Olaf Hvasshovd, who also fills the role of councillor. Svein-Olaf is a professor at the institute of Computer Science at NTNU Trondheim, and has a lot of experience with sheep husbandry.

1.4 Why, what and how: Objective, deliveries and methods

1.4.1 Objective

The main objective is to deliver a report that describes the task, related projects, the final product that was developed based on the task description, and the results from the focus group testing.

First the requirements of the application must be extracted from the task description. A basic design can then be created based on the requirements. The combination of these two will determine how the alpha version of the application will be programmed and designed. This alpha version will then be iterated over and further modified to fit the requirements of the contractor. Lastly it will then be tested by a focus group. The results from this testing along with the internal testing done by contractor and creator, will be used to draw a conclusion about the current state of the application, and what needs to be implemented/changed in a future version.

Main Goal Create a mobile application that fits the provided specifications, test the product internally with guidance from the contractor, and then lastly run usability and user experience tests with a focus group.

- **Intermediate objective 1** Research what solutions currently exist that are related to the problem at hand.
- **Intermediate objective 2** Determine what is required for the application, and do research into systems that can be implemented to achieve these requirements.
- **Intermediate objective 3** Create an alpha version of the application to run internal testing on.
- **Intermediate objective 4** Iterate and improve the application based on feedback provided by the councillor until his requirements are met.
- **Intermediate objective 5** Create a focus group, prepare and run usability / user experience tests.
- **Intermediate objective 6** Present results from the analysis of the focus group testing.
- **Intermediate objective 7** Draw a conclusion as to how well the solution works (in terms of usability and user experience), and determine what needs to be addressed in future versions.

1.4.2 Plan

September 17th, 2019: Finish the design of the application and the preliminary research.

December 31st, 2019: Program the alpha version of the application.

February 11th, 2020: Test the application internally with the contractor, and also begin working on the final report.

May 1st, 2020: Run focus group testing and compare the results from the testing.
June 1st, 2020: Deliver final complete version of the report.

1.4.3 Methods

The methods used for this project will be the following:

Research Research current relevant mobile applications, and what systems can be implemented for this project.

Implementation Creating the application based on the research.

Internal Testing Internal testing between the creator and the contractor.

Focus Group Creating a focus group.

Testing Running tests on the focus group.

Analyse Analyse the results from the testing and form a conclusion.

1.5 Report Structure

This report consists of 9 main chapters, with several sub-chapters. These chapters are as follows: introduction, state of the art, research, requirements, implementation, testing, results, discussion and conclusion.

The introduction introduces the author of the report, the contractor, the task, and what this thesis is trying to achieve.

The state of the art chapter contains what solutions already exists and give a rundown on how these work and compare their functionality to what this project is trying to achieve.

In the requirements chapter the report will go a little further into detail what features are required of the application, and why they are required.

The research chapter looks into general mobile systems, as well as relevant systems that can be used to fulfil the requirements of the contractor.

Implementation goes over the original design of the application, how each of the systems that have been selected to implement in the application works and what functionality they provide. Lastly it goes through the application page by page describing how it functions.

In the chapter for testing, the report will describe the testing method in detail, as well as how the different tests will be executed.

The results chapter then looks into the results from the tests run in the previous chapter.

In the chapter for discussion will contain the results from testing will be discussed, how the testing process itself went, general discussion about the applications features, and all thoughts around the application and the report.

Lastly, the conclusion chapter will give a concise conclusion of the results from the entire project, with notes on future development.

Chapter 2

State of the Art

This chapter will look into the current state of applications that have been determined to be relevant to this thesis. It will also provide comparisons between these and the application described in this thesis.

2.1 Findmy

Findmy is a Norwegian firm that produces equipment for electronic tracking of animals, humans and objects. What separates them from the rest of the flock is that they do their tracking with low-orbit satellites, and therefore they are not dependant on cellphone service. The creators (or gründers if you will), are Norwegian sheep farmers that know the challenges of monitoring sheep in rough terrain and designed this solution after they themselves lost roughly a quarter of their lambs over a single season. Currently they have roughly 40 000 units in active usage [8]. Their units are not only used on sheep, but are also to track humans, boats and large machinery. They can also be adapted to be used on reindeer and camels.

2.1.1 Findmy Model 1

The Findmy units are relatively small and fit well on the collars that most sheep already use (see fig 2.1).

Findmy recommends that you cover at least 25% of the flock as to provide a good overview over your animals. They also provide a calculator to allow you to determine how many bells are correct for your usage [9]. The bells are currently priced at 1890,- NOK excluding taxes, which can be lowered upon bulk purchasing.

In addition to the bells they provide what they call a charging board (fig 2.2), which can charge 5 bells at a time and takes roughly 2.5 hours to fully charge one set. A full charge will last just about one season, if the unit is set to ping their location about once per day. The charging board is also used to update software and to set the plan for how often the unit will ping home. They also advertise that neighbours can share a charger, thus bringing the cost down further. A single

Figure 2.1: Findmy tracking unit [9]

Figure 2.2: The charging board for the Findmy system [38]

charging board costs 4500,- NOK excluding taxes. There is also a yearly user-fee that the sheep farmers have to pay, and this comes out to 229,- NOK.

2.1.2 Findmy Model 2

Findmy has recently released what they call the model 2, which is the next generation of bells that they have designed. These are advertised to have longer battery life (up to 2-3 seasons), be lighter than their predecessor, not require a dedicated charging board as they have swap-able batteries, and the whole setup can be done via your mobile phone. It also tracks a lot better in difficult terrain.

2.1.3 Expansion

Figure 2.3: Findmy being implemented on livestock in Kenya [17]

Findmy is now also providing their units to Kenya to help with their tracking of livestock [17] (see figure 2.3). They're starting with 10 bells, but hope to expand and eventually setup a department locally. Findmy has also worked with tracking camels in the Saudi-Arabia gulf, and even livestock in Brazil.

2.1.4 Alarms

The unit also allows for several alarms which are as follows:

Movement alarm: The movement alarm allows for notification to the user if the animal has not moved 40 meters in the last 48 hours based on geodata. In addition to this, units starting from 2016 also has movement sensors built in, which checks every 3rd hour if the animal has moved, and if it has not it will notify the owner.

Geofence: The geofence allows the user to set up a virtual fence on a map, and the user will then be notified if the animal wanders outside of this fence.

Turmoil alarm: The turmoil alert uses a system that can identify when a group of sheep has moved in a radical manner in comparison to what is normal. This would indicate that the flock could've been attacked by wolves or bears.

2.2 Telespor

Telespor is a Norwegian company that uses LTE-M and Narrowband IoT technology to provide their users with location data of their animals. They currently operate in Norway, Sweden and Iceland [34].

2.2.1 Radiobjella

Their latest unit, which is named Radiobjella (see fig. 2.4), is the 4th generation of their unit as of the time of this report, and more specifically uses GNSS technology for getting the location of the unit. Their unit offers two-way communication, which allows the user to update in real time how often the unit shall ping it's location, the sensitivity of the alarms, and also whether or not they are activated at all. These alarms are covered in the next section. The unit's battery is replaceable and it is recommended to change this after each season. One of their units with a battery costs 899,- NOK excluding taxes, however they do offer bulk buying which lowers the price depending on the amount of units that is purchased. A season pass which lasts for 5 months can be purchased alongside a battery for 140,- NOK.

Figure 2.4: The Radiobjella unit [35]

2.2.2 Alarms

The unit offers multiple alarm settings which can inform the owner of information that is important and time sensitive. The four alarms it offers are as follows:

1: The animal has not moved in the last three hours.
2: The animal has stayed in the same position over a longer period of time.
3: The unit has been unable to transmit it's position the last two pings.

4: The unit is low on battery.

The sensitivity of these alarms can be modified to accommodate an animal that is less active.

2.3 Nofence

Nofence is a Norwegian company that offers a system that has a solar-powered GPS collar, and a digital map. Users can create virtual fences in the application that is used to control where the animals are allowed to go. If/when an animal crosses the user defined virtual fence, the collar will start beeping. If the animal continues walking away from the fenced area, the animal will receive a small electric shock. If the animal continues even further, the system will not shock the animal further to prevent harming it.

2.3.1 Unit

The nofence unit (fig 2.5) is a self contained system with a small battery port, a GPS receiver, Bluetooth transmitter, accelerator sensor and new LTE CAT-M1 cellular technology with 2G fallback [25]. A nofence collar unit costs 1850,- NOK. The season pass price varies greatly with the amount of units that is being used, but the base cost is 6,- NOK per collar/day.

Figure 2.5: The nofence unit [25]

2.3.2 Battery Pack

Nofence offers extra battery banks (fig 2.6) that are easy to click in and out from the collar housing, making it easy to swap batteries when it's needed. The batteries are the same type that is used in Tesla cars [25].

Figure 2.6: The nofence external battery pack [25]

2.3.3 Nofence Beacon

The nofence beacon (fig 2.7), is used to temporarily deactivate the GPS receiver in the collar units. It is used to prevent reporting of inaccurate positions when the animals are inside, or in other areas with poor GPS coverage. As a side effect, this will also help save battery power as the unit doesn't use the GPS receiver as much [25].

2.3.4 Battery Charger

Nofence offers a standard European wall charger for their battery units (fig 2.8), and they have also have adaptors available for non-European sockets.

2.4 Shiip

Shiip is a Norwegian company that mainly focuses on what they call B2B-sales (Bonde 2 Bonde), and they work closely with Telenor and Telia in Norway to

Figure 2.7: The nofence beacon [25]

Figure 2.8: The nofence battery charger [25]

provide users with flexible methods of tracking sheep, other animals, and miscellaneous objects [30].

2.4.1 Unit

The shiip unit (fig 2.9), much like the Radiobjella unit from Telespor (see chapter 2.2) also use GNSS technology for accurate location data, and Narrowband IoT for communicating with the servers from the units. The unit itself costs 799,- NOK, and can be lowered with bulk purchase. A season pass for the usage of the units cost 99,- NOK, or one can purchase a full year for 233,- NOK.

They claim to have a max battery life of 10 years under optimal conditions,

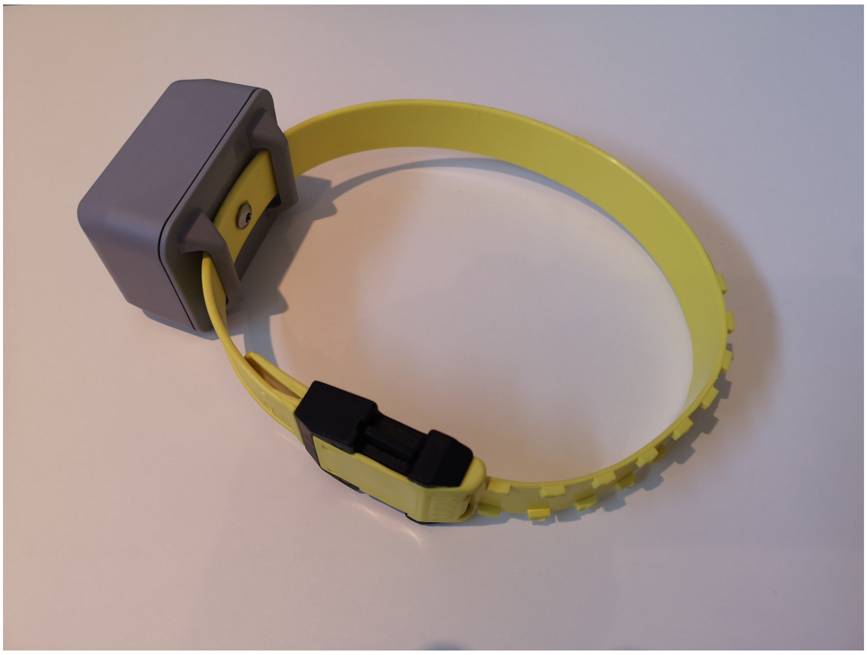


Figure 2.9: The shiip/smartbjella collar unit

however in a more realistic situation they expect the battery to last somewhere between 5-8.5 years, depending on conditions and how often the user sets the device to ping home. Shiip also advertises for indoor tracking, however they do not go into further details on their product page [29]. They also offer two way communication with their device. The most important feature is that they can provide an unlimited data plan (depending on usage and reporting intervals). However they do not directly sell their product, instead this is done via Smartbjella (see chap 2.5).

2.5 Smartbjella

"Smartbjella" is a Norwegian system created by a collaboration of demanding sheep farmers and experienced technology providers [31]. Over the years they've tested out several different units designed for tracking sheep and reindeer based on GPS technology, and so they have actually tested several of the units that has been briefed about earlier in this chapter (namely Findmy and telespor). They started with testing something called "Dødelighets sendere" for two years [32], these units were a part of a test project that had been cancelled and so Smartbjella took over these units for their testing. This system used its own radio system, so the sheep farmer would have to go out with an antenna and ping their way to the animal. This required the sheep farmer to regularly take trips out in the field and so eventually that became the reason why they stopped using it. They soon switched over to Telespor, and started out with testing 10 units back in 2007 [32], however they found that the earlier models would often need to be completely replaced and so it became a little too expensive for them in the long run. From 2014 they started using Findmy, but these units were also quite expensive, and it was a lot of work to recharge and configure the units so they ended up moving away from this product. In 2018 they discovered Shiip, they didn't actually get any units directly from them in the beginning to test with, but they really liked the technology that they had been using for their units and so when they decided to make their own units they took a lot of inspiration from them.

The unit that they ended up with was designed in cooperation with Shiip and can be read about in chapter 2.4.

2.6 Features, Alarms, and Price Comparisons

In this section, the thesis will compare the systems that have been discussed so far based on their alarms, prices and features that they provide. The first table is an overview of what company the different units belong to (table 2.1), while the three remaining compare the values that has been previously mentioned.

Name	Units
Findmy	Model 1 Model 2
Telespor	Radiobjella
Nofence	Collar
Shiip	Smartbjella

Table 2.1: Overview of the units that each company offers.

An important note to notice in table 2.1 is that Findmy has two units listed. This is done as it was not quite clear where the line between unit 1 and unit 2 was. There are some notes of features that the model 2 has that the version 1 does not, but when it comes to ordering the units it is hard to determine exactly what unit is being ordered.

Units	Unit Price	Season Price	Charging Board Price
Model 1 Model 2	1890,-	229,- Yearly	4500,-
Radiobjella	899,-	149,- Season	N/A
Collar	1850,-	6 per collar/day	N/A
Smartbjella	799,-	99,- Season 238,- Yearly	N/A

Table 2.2: Price comparisons of the sheep tracking units/systems.

In table 2.2, one can see that the prices per unit is quite different, with the Findmy models and the Collar unit from Nofence being the most expensive units to purchase directly, with Smartbjella being the cheapest. In addition to the cost of the units, Findmy also has a charging board for the model 1 units which has to be purchased separately and costs 4500 ,- NOK.

In terms of the seasonal pricing, Smartbjella is the direct winner of paying only 99,- NOK per season, but they offer a less amount of features and alarms than a lot of the others. If one intends to purchase a year pass however, Findmy pulls ahead of Smartbjella with only a 9,- NOK difference.

<table>
<tr><th>Unit</th><th>Features</th></tr>
<tr><td>Model 1
Model 2</td><td>Different map styles
Sauekontrollen cooperation
Low-orbit satellite tracking
Loss of lamb map
Tracking logs</td></tr>
<tr><td>Radiobjella</td><td>GNSS Location
Two-way communication
SMS Notifications
Waterproofing</td></tr>
<tr><td>Collar</td><td>Nofence beacon
Battery packs
Solar powered
Goefence
Electric shock</td></tr>
<tr><td>Smartbjella</td><td>Two-way communication
GNSS location
Up to 10 years battery life</td></tr>
</table>

Table 2.3: Feature comparison of the sheep tracking units/systems.

In table 2.3, one can see all the different features that each system offers, which has been extracted from the earlier information regarding each system. The table makes it quite clear that while some systems have similar functionalities available, a lot of them also have features that are unique to them. In order to determine what system would be the best choice here, one would have to take into consideration how important each feature is for your usage. Different farmers will probably have different priorities in terms of what they expect a system to be able to deliver, which is possibly also why they have such a large amount of different available features.

Unit	Movement	Goefence	Turmoil	Stationary	Battery	Transmitting
Model 1 Model 2	X	X	X			
Radiobjella	X			X	X	X
Collar		X				
Smartbjella						

Table 2.4: Alarm comparison of the different units/systems.

In the table regarding alarms (table 2.4), one can see that Radiobjella has the most amount of alarms available to the user, while Smartbjella offers no real alarm system functionality. Alarms can be quite valuable to the sheep farmer as some information is highly time sensitive and so a farmer needs to be alerted quickly in order to avoid disaster. For an example, the turmoil alarm can alert the farmer that there are predators near their sheep flock.

2.7 Allma

Allma is a company that delivers a service that is quite different from the others that have been discussed so far in this chapter. While it is a quite different field of work, there are striking similarities between Allmas solution and the application developed for this thesis, as will be described further in the discussion section of this chapter.

Figure 2.10: The Allma application on different platforms [2]

Allma provides their users with a complete forestry planning system [1]. Basically it allows you to register the area of forest that you own, and to monitor this with an application that is available on mobile devices and PCs (fig 2.10). Users can plan out planting of new forest, or what parts of the forest they want to chop down. This can all be done from either their home our out in the field by using a mobile version of the application. With this tool the user will have an excellent overview over their forest, and it will also help you determine when the most profitable time to sell wood is.

2.8 Discussion & Conclusion

This chapter has mostly been about different technologies that help you directly track the animals without the need of being in the field. This is quite different to what this thesis is trying to achieve but it is the most relevant as they both strive to ultimately achieve the same goal, which is to make it more efficient to look after animals. This in turn will also make it more profitable for the sheep farmers as they can spend less time on mundane tasks.

Overall it is hard to determine which of the sheep based systems is the best, as it really comes down to what features and alarms are important to the way a farmer wants their system to work.

In terms of direct comparison however, Allma - which is about forestry, is the closest to the solution that this thesis is describing. The Allma mobile application allows the user to register the status of their forest as they are out inspecting it. The difference between that and the application that has been designed for this thesis is mostly the method of registration, and that Allma doesn't continuously track the user's position in the field as this not important for their functionality.

Chapter 3

Requirements

In this requirements chapter the author will go into detail about the different requirements that are set for the product. These requirements have been developed in close cooperation with the contractor. The chapter is split into two major sections, where the first is regarding the usability of the application and the second is about the offline capabilities that are mandatory for the required functionality to be achieved.

3.1 Usability

This section will go into what features are necessary to be implemented into the application for it to be a valid resource for sheep farmers.

3.1.1 Recording User Position

In order to keep track of where the inspection tours have been, the person in charge have to record their location in some manner. The accurate recording of this data is important as to allow the farmer to know what parts of the map has already been checked out and where the sheep have been observed. The sheep can also move across large tracts of land, which means that several inspection tours have to be taken in order to cover the map and to register all the sheep and their locations. Sheep farmers also often have helpers to do this for them as the amount of inspection tours can become quite overwhelming. The data from these trips are quite important as this allows for better planning of future inspection tours in later seasons, while also giving a good idea where the sheep are heading and where they tend to flock to while they're out grazing. In order to achieve this, the application will need to continuously record the GPS location of the user as they traverse the land. The interval between each recorded position should not be too long as to allow for a detailed path to be formed, and also not too short to avoid unnecessary amount of function calls, which will drain more battery life without providing any noticeable improvements in accuracy.

3.1.2 Recording Sheep Observations

When the user performing the inspection tour sees a flock of sheep, they will need to register this with the application. The user should be able to view a map which shows their current position and allows them to select a location on the map that the user observes the sheep to be in. After determining exactly where the sheep are located, the user can then save that location as an observation. The observation will automatically also record the user's location at that moment. It will also store information such as the distance to the sheep (calculated by the location of the observation and the user's position), the time that the observation was done, and also preferably an image that displays both the user's position, and also the observation's position.

Usually when one does this, one tends to differentiate between relatively short distance observations (less than 200 meter distance) and longer distance observations (longer than 200 meter distance). The reason one does this is because at a shorter distance it is possible to not only see better how many sheep there are, but also what is on their ear tags which can tell the user who they belong to. On the long distance observations it is more important to only register the amount of sheep one can see, and hopefully also how many lambs there are. The ewes can also be marked with a tie (around 10 cm x 3 cm in size) that has a colour code corresponding to how many lambs that ewe has. This information is important, as the sheep farmer can calculate how many lambs are supposed to be in the flock and compare it to the actual amount that they see. This will give the sheep farmer an early heads up about losses of sheep early in the season, possibly due to predators in the area. This data can be useful to get the sheep out of the area where the predators are at, or in the future prevent them from going there in the first place.

However in this prototype version there is not a distinction between these, although it could simply be added by using the distance that is already stored to display a method of registration suitable for that distance. With a possibility of an override if conditions require a different method of registration (e.g bad weather gives poor visibility).

3.1.2.1 Same Flock Observation

A user might observe the same flock of sheep multiple times during an inspection tour and so the application will need to be able to record having seen the same flock again. This will be achieved by linking the data for the new observation to the old one, and thus if one wants to change the number of sheep that has been observed previously, one would only have to edit the first observation. So with this function, one can save multiple locations of the same flock and patters in their movement can be extrapolated.

3.1.2.2 Observations With Binoculars

A problem with registering sheep is that if the user attempts to get close to the herd, the herd will most likely run away from the user. Thus one has to observe them in most cases with binoculars from a distance. This can make registration of sheep tricky as the user will have to keep their eyes in the binoculars to make sure they don't count the same sheep twice as the sheep move about. It is highly important that the registration process can be achieved with a single hand and without the need for the user to observe the screen. Preferably the user should also be able to determine how many sheep they have registered at any given time, again without observing the screen.

3.1.3 Recording Wounded Or Dead Sheep

If a user finds a wounded or dead animal on the inspection tour then they should be able to register this as it is highly important. The application should be able to store where the wounded or dead animal was found, what was done with the animal after its discovery, who the animal belongs to, and if the animal is dead - try to determine the cause of death.

This has not been implemented in the application as the focus of the thesis is the registration of groups of sheep. This should not be very difficult to add in later version, as it would take a lot of the same code from a normal observation, and just add a few text fields to fill in the relevant information. Possibly also implement the camera of the mobile device to be able to take pictures of the sheep in question.

3.1.4 Recording Other Observations

When a sheep farmer is out on an inspection tour they might observe something that is relevant, but is not a direct sheep observation. This can be that the user has observed a predator, observed other animals in the area, or what ever the user might find to be relevant. These observations should also store the location of the user and the observation.

In the application developed for this thesis, the other observation has not been implemented as it is not an essential functionality. However this could easily be achieved by using most of the methods used for sheep observation, and then in addition have an input field for text that the application will store.

3.1.5 Map Accuracy

For the application to be of any use the maps have to be accurate. This means that the map information needs to be detailed enough so that a user can determine exactly the location of the sheep on the map based on what they can see in the environment around them.

3.2 Offline Capabilities

The application needs to work offline as it is not always a given that a user will have cell reception while out on the inspection tours, and so in the next subsections the thesis will describe the different offline functionalities that are required for the application.

3.2.1 Storing Maps Offline

As the application should be able to be used offline, the maps have to be able to be downloaded in advance before a user goes out on an inspection tour. This map will also necessarily have to function like a normal map when in use and allow for data to be displayed on it (such as user position, movements so far, observations so far, etc). The user should also be able to remove old maps in order to allow the user to save storage space on their device in the future.

3.2.2 Save Data & Data Synchronisation

There is a lot of data that the application is creating while the user is out on an inspection tour and so this needs to be able to be stored locally on the device. This data should then, once a data connection is available, synchronise itself with a storage server of some sort to allow for long term storage, availability on other devices and as a way to backup the data.

Chapter 4

Research

In this chapter, the report will look into relevant systems that can be used for the development of the application. Firstly it will look into what options are available and what exactly they offer. Afterwards a selection will be done based on what is most relevant and useful for this prototype.

4.1 Mobile Solutions

Figure 4.1: Mobile solutions

4.1.1 Android

Android is an open source operating system for mobile devices, which is provided by Google [4]. As of 2019 they hold 86.1% of the global market share for mobile operating systems [18], and so it is a very relevant operating system to develop for. Android has a lot of freedom when it comes to developing for it, as Google is taking a more hands off approach for developers and putting the responsibility in their hands. Google provides excellent guides for how to create different Android modules (e.g. how you implement the standard Android navigation) and as it has a large user base quite often someone else has encountered the errors you might encounter while programming. As Android is open source and widely used, there's also a lot of 3rd party developers that have written open source APIs/SDKs that are available to be used in the development of your own application. Some of these cost money but there's also a lot that are free. Google also provides a lot of services that are made specifically for mobile devices (not necessarily just Android), and these will be discussed further down in this chapter. Posting applications on the app store is available for everyone, which makes it easy for anyone to make an application, to post it and possibly earn some revenue. However this also adds a security risk as people with malicious intents may upload some form of malware.

4.1.2 iOS

iOS is an operating system that Apple develops for their mobile devices. As of 2019 they hold a market share of 13.9% for mobile operating systems [18], and so it's not as wide spread as the Android equivalent, but none the less it is a large share of the market. iOS is in general more strict with development, as in order to get the application on the market store it has to go through an approval process. This can make it hard to develop for the iOS market, however as the application has to go through the aforementioned approval process, the app store as a whole is safer. Apple also provides guides on how to create basic modules [5], however as the market share is smaller and the process is more strict, the amount of 3rd party developers is in general smaller than that of Android.

4.1.3 Web-based Application

Web-based applications are applications that are created to be run in a web browser environment. Because of this they are independent of what operating system the user has on their device, which means the application is very versatile. However as one is designing it for a browser, one cannot naturally use any of the built in functionalities of the devices themselves. There's also a lot of limitations when developing for a browser as your application will essentially be a webpage, and thus can only perform the actions of a standard webpage. This can limit overall functionality and usability of the application. Developing such an application with offline capabilities in mind seem very difficult, if not near impossible. One can attempt to keep things in memory until a data connection is available, but this does not seem like a very good option.

4.2 Maps

Figure 4.2: Map Services

4.2.1 Google Maps

Google maps is a map service provides by Google. It allows a user to view where they are currently located in the world with multiple different views (2D map, 3D map, satellite map, street view, etc). It also allows for navigation from point A to point B. As it is a service provided by Google, it has a lot of documentation on how to implement it for Android. Google Maps also has a lot of miscellaneous services that for an example allows a user to search for nearby restaurants, but that is not so relevant to this paper. With the Google Maps SDK that is available for Android, one can add these maps to your application. This allows you to have the same navigation methods and gesture control as with the normal Google Map app, and in addition also allows for you to draw your own lines on the map, and to add custom/images on top of the map. One can also download sections of the map for offline usage [16].

A downside with Google Maps is that the application is mostly focused on navigation in cities and not so much for the Norwegian outdoors, and so the maps are not very detailed and not realistically very usable for this project. However as one can add tile overlays, one should be able to overlay it with a more detailed map from kartverket for an example (see chapter 4.2.3 for more information).

4.2.2 MapBox

Mapbox SDK is an open source project that is free to use for smaller scale projects and provides a whole host of services related to maps and navigation. The free version supports up to 25 000 monthly active users, 50 000 monthly loads of the map and 50 GB of storage for map tiles [21]. As this project is a prototype, this is well within reason. Mapbox has a lot of the standard map functionality that users are familiar with from other similar map applications, namely different styles of map, camera manipulation, gesture control, map pins etc [23]. Mapbox also allows for saving maps locally to a device, thus allowing for offline usage. Mapbox also provides a studio which allows users to create their own map styles [22]. This gives a user control on how the map looks like on all devices quite easily from the mapbox webpage.

Mapbox had a slightly more detailed map for the outdoors in Norway, however this too came up short as to what was required for the application. Luckily one can also add tiles to these maps from the mapbox studio page, and thus one can get detailed maps that are up to date (from kartverket from an example, see chapter 4.2.3).

4.2.3 Kartverket

Kartverket is the official Norwegian mapping service, and provides national geographical information for both private and official users [28]. This means that they keep up to date maps of Norway and provide these for free for anyone who wishes to use it. They also provide multiple web pages and applications that already have these updated maps implemented in them. This map data is available to request and download from Geonorge [13]. Basically for smaller files one can directly download them, however if you wish to download a large amount of data you have to send in a request with your email address to the system. After the request is processed, a link will be sent to your email address which allows you to download the requested data. There's a large amount of different map types available for download, however for this project the map service that is selected will be requiring a tiling layer to achieve a map with enough details and so only the GeoTIFF files are relevant.

4.3 Storage

Storage is an important feature of any given program, and for mobile devices there are several valid options that could be implemented. Some even are directly built into most mobile devices already, such as the SQLite database, which will be described in the next section.

4.3.1 SQLite

Figure 4.3: SQLite logo

SQLite is a lightweight SQL database engine. It is the most used database engine in the world and is also built into all mobile phones [33]. It operates much like any other SQL based database, where one defines how the database should appear and how the data is structured and stored, in this case with a schema, before then writing/reading from the database using the parameters set there. This does not allow for easy storage of files, and also one would have to create their own server service in order to have this data available online.

4.3.2 JSON

Figure 4.4: JSON Logo

JSON (JavaScript Object Notation) is a data storage format that is designed to be both lightweight and easy to work with for both humans and machines [20]. In its essence, JSON is text that has been written with JavaScript object notation. Meaning it is a list of information that has been structured in a way that is very readable and easy to parse. In order to make use of JSON for a mobile device, one would have to use a standard parser to convert the instances of classes into data that can be stored whenever changes to the information is made, or the application is closed. This data would also then need to be parsed back into instances whenever the application is started up again.

4.4 Google Firebase

Figure 4.5: Google Firebase logo

Google Firebase is a set of services that Google provides for application developers, these are free to use and also scale well with larger applications [15]. Although the application that this thesis describes will remain as a small scaled prototype, it is good to know that the services that form the backbone of the application could handle a large user base.

The reason why this set of services are particularly interesting is that they work mostly as well offline as they do online. Basically the device with the application hosts a version of the database locally, and then any changes made to this database will be uploaded/synchronised to Google as soon as the user is online.

Particularly interesting for the development of this application is their Cloud Storage service, Cloud Database service and their authentication service, which are described in further detail below.

4.4.1 Firebase Cloud Database

Google Cloud Database is a NoSQL database, meaning that it is not as strict as other traditional database systems in the terms of how data is being stored, and allows the user to essentially use it as a database of objects. This makes it very easy to implement it in an Android application, as both Java and Kotlin are object oriented programming languages. The way it works is that the database is set up as a set of collections and documents. Documents in this context would be referring to objects in the code, while collections would be a method of storing multiple objects (sort of like an array of objects). Thus users can interchangeably move objects to and from storage.

4.4.2 Firebase Cloud Storage

Google Cloud Storage is a sub service of the Firebase family that allows for storage of files. For this project this could be valuable as a method of storing the images of where the observations took place online and thus allow the user to download these images at a later date. Potentially if one wants to add images the user took themselves, this could also be uploaded to this system.

4.4.3 Firebase Authentication

The Firebase authentication is a service that makes it very easy to create a fully functional login system. As all login is being handled by Google which has many years of experience, this should make the application in general safer. Using this in combination with the cloud storage, one can easily separate out what objects/files belong to what user.

4.5 Comparison

In this chapter multiple systems have been briefly discussed, however some of them are more relevant and seemingly interesting to work with than others and so decisions had to be made.

In terms of operating system Android was selected as it has an overall larger market share, and thus more users and developers meaning more resources to aid with the development. It is also more accessible to design for as the author has a mobile device running Android. The author also has previous experience with Android programming, and naturally that knowledge will come in handy. A web-based application would also be a good option as this would allow for cross platform usage, however it has too many limitations and would be difficult to implement in a way that allows for offline usage. It should be noted that the following services in theory could also be used on iOS devices and on web pages, thus allowing for a larger coverage of the market. In the original concept the idea was that there would also exist a webpage that the sheep farmer would be able to use to create reports automatically, but more on this in the discussion chapter (chap 8.5.6).

For maps the plan was originally to use Google Maps services, however as this was discussed with the councillor for the thesis it became clear that the maps were not nearly detailed enough. Which lead to MapBox which was slightly more detailed. Mapbox also supports the major features that Google Maps offer and so they were a good option to go with. As development went on, it became clear that one could overlay the maps with official maps from Kartverket, and thus a lot more detail could be added. Mapbox also has easy methods for allowing the storage of maps locally, which is a very important feature for this project.

Google Firebase was selected to be used for the application's storage, database and authentication as this was an all in one package that offers excellent options for both offline and online usage. Also implementing these different systems appear to be relatively simple to do.

Chapter 5

Implementation

This chapter will describe the method of implementing the requirements previously made in the thesis. First it will go into the original design and a little bit about how that changed, before going into detail of each system that was included in the development of the application, then lastly describing exactly how the application operates in its current state in each activity of the application.

5.1 Original Design

The original design was first created with pen and paper, but as it looks extremely rough the report will show the first proper designs that was created in draw.io [7]. These designs were created very early in the development phase, before any real programming work had started as to give the author a good idea of what the product should look and operate like.

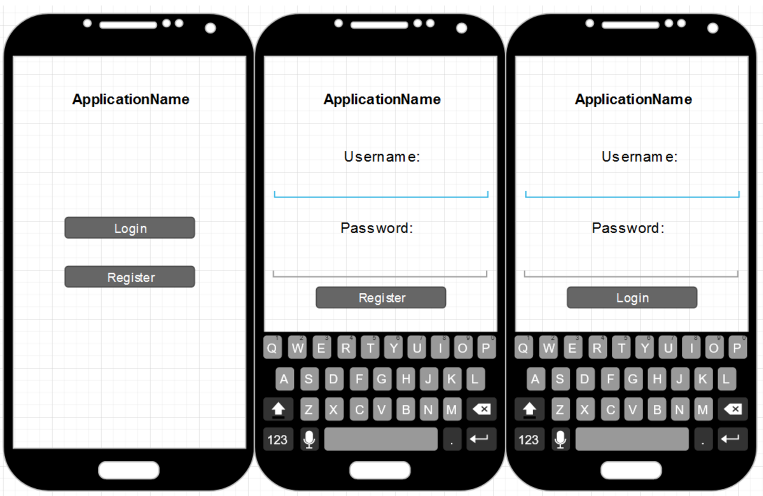


Figure 5.1: The original login design.

There's not too much to say about the original login screen (fig 5.1), compared to the final result (fig 5.14) they look quite similar. However the final version is a lot more colourful than the original design and looks a lot less rough.

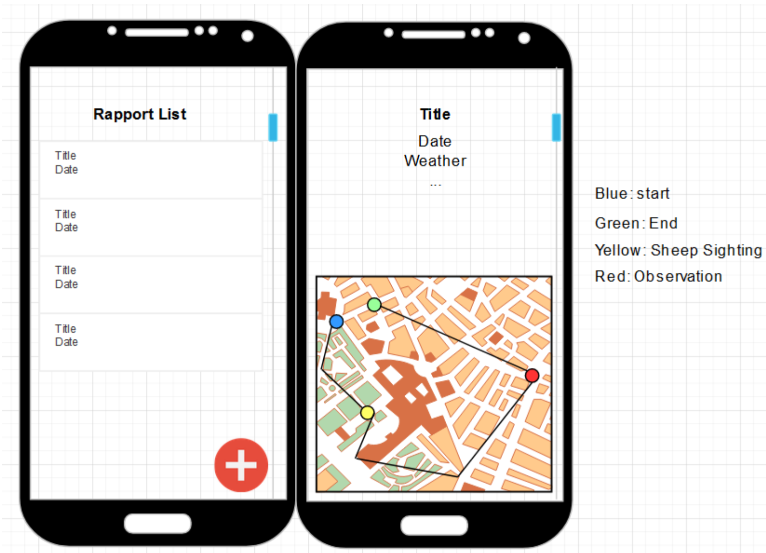


Figure 5.2: The original report list & overview design.

The original report list and overview (fig 5.2) are also not too different from the final product. The report list would naturally look the same as it is simply a list of the reports that have been produced with the application. The overview was a little bit different in the final product, as the map has been moved to a separate window as to prevent having too much clutter in one page (fig 5.15, 5.17), and to split up functionality to not overwhelm the user.

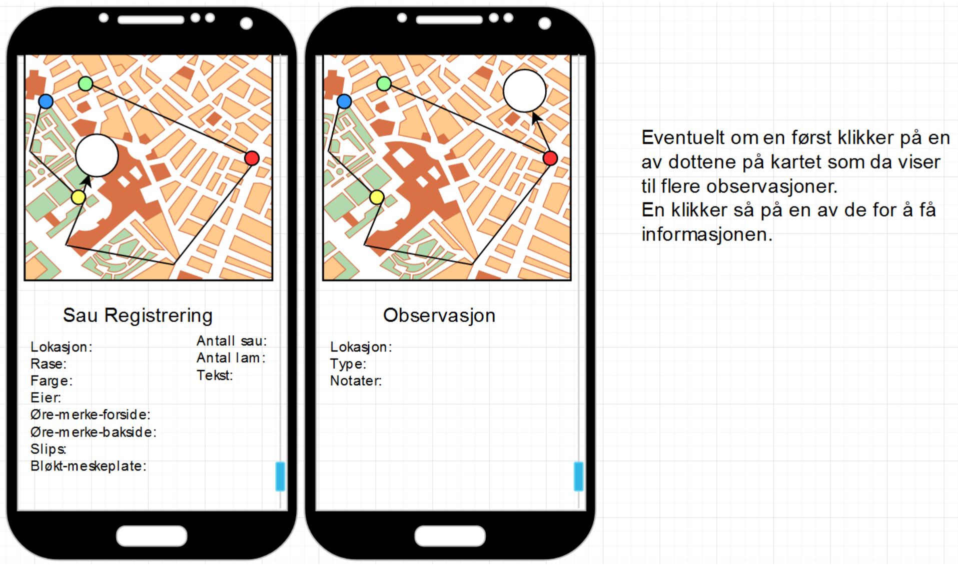


Figure 5.3: The original observation overview design.

The observations was also thought to be a part of the same page in the original plan (fig 5.3), this was however changed to be its own activity of sorts. Instead one can navigate to the observations from the map as in the original plan, but it will then redirect the user to the observation page.

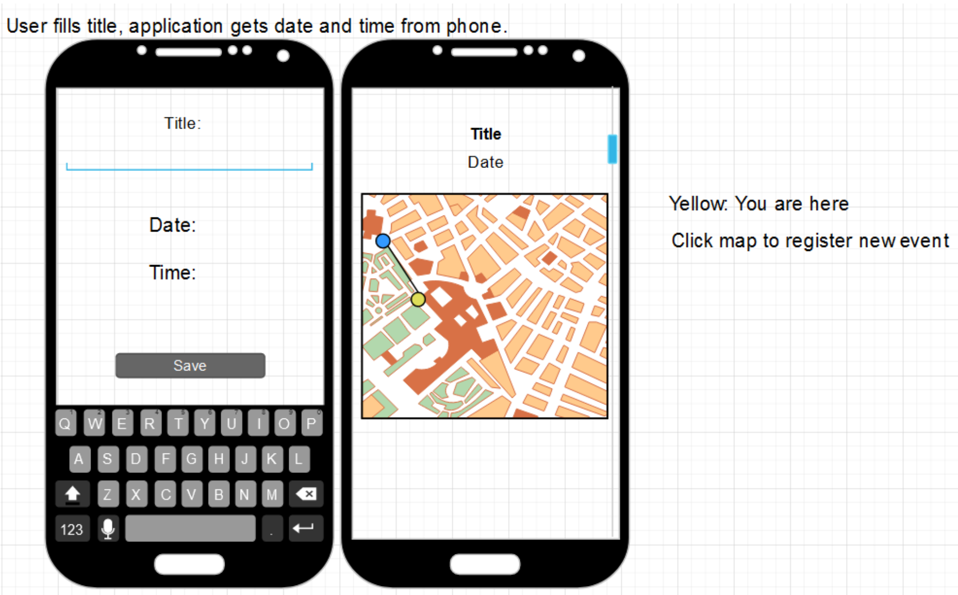


Figure 5.4: The original report creation design.

The original report creation is not a lot to comment on (fig 5.4), basically the user would input a name for the new report, and then the device would collect the date and time automatically. This is what the application does in its current state as well.

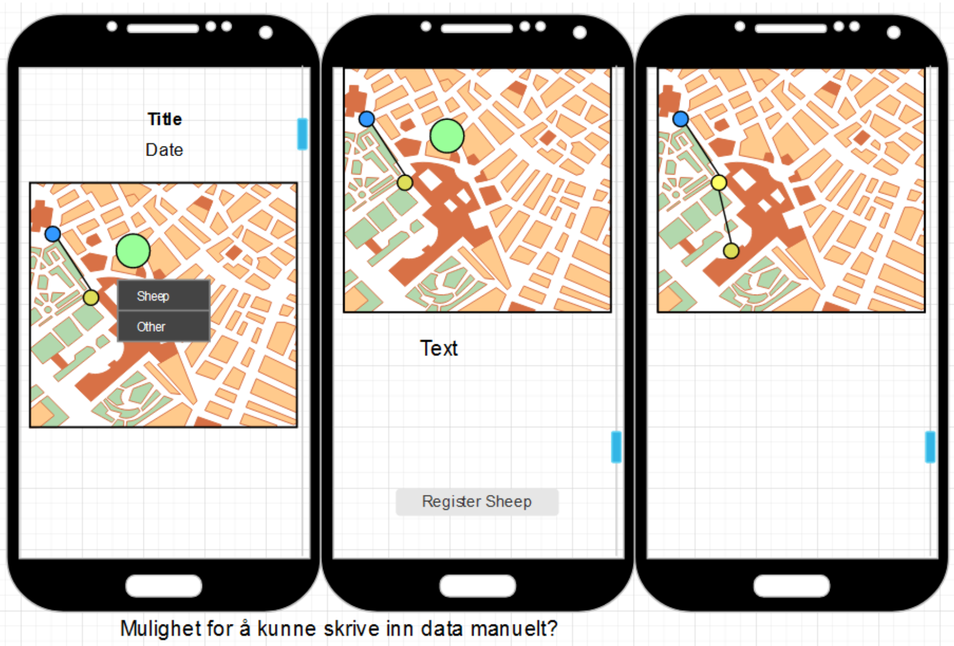


Figure 5.5: The original register new observation from report design.

The original method for starting a new observation is also quite similar to how it works in the application (fig 5.5), the user here clicks on the map to initiate a new observation, this is exactly how it works in the latest version as well, with the details of the observation being made in a separate activity.

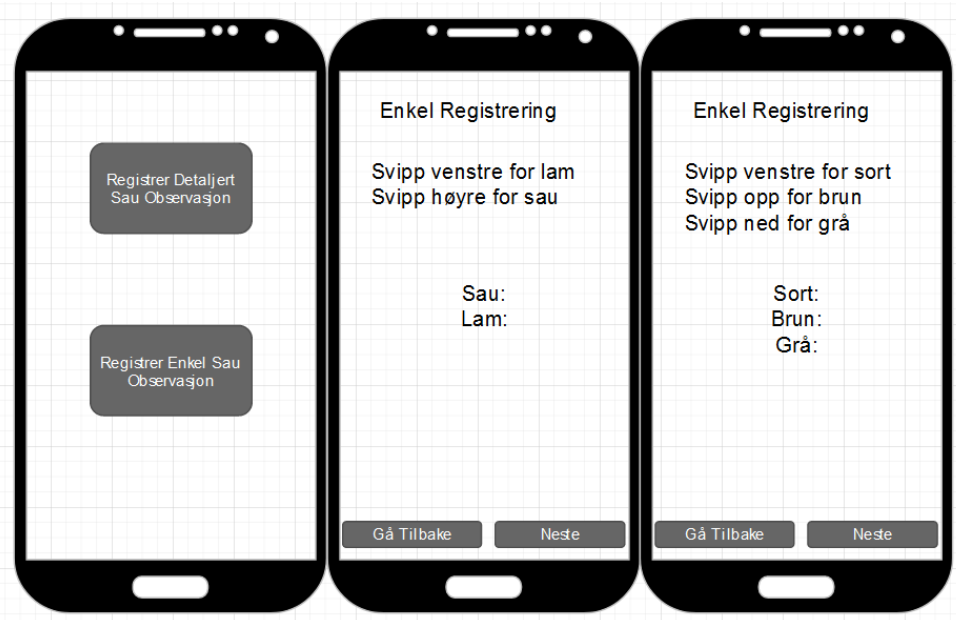


Figure 5.6: The original register selection & simple registration design.

The design was first going to have two methods of registering sheep, which would be a detailed version and a simplified version (fig 5.6). The simple registration would basically only register adults and lambs, before then allowing the user to divide them up into the colours of their wool.

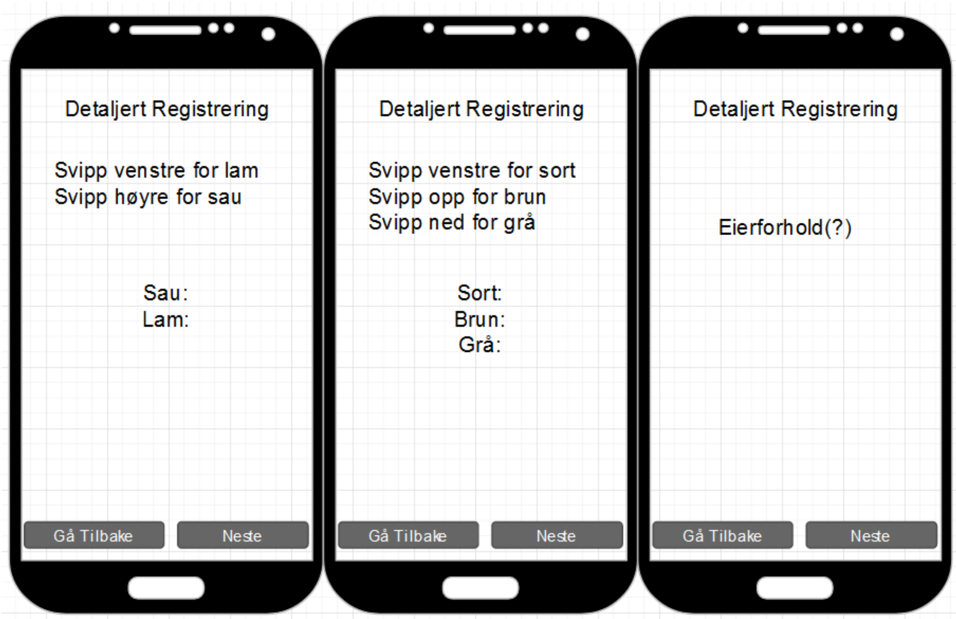


Figure 5.7: The original detailed register design.

The detailed version of the registration would not be too different from the simplified version, the only exception here was that it would have another page for determining who the sheep belonged to (fig 5.7). This is not very detailed in the figure as the author was unclear what was meant by this.

5.2 Android

As stated in the previous chapter, the author elected to go with the Android operating system. When programming for Android devices, it is typical to select a minimum version of Android that the application should support. This is done as a lot of the Android user base is split up between the different versions (see fig 5.8).

For this project the selected minimum version is set to version 4.4 (also named KitKat or API level 19) as it would cover about 98.1% of the Android market (see fig 5.8). Going back much further than this does not drastically affect the amount of compatible devices and thus there is no real reason to do so. It would also reduce the amount of available features to use in the application.

5.3 Android Studio

Google provides a free to use programming IDE for developing Android applications named Android Studio [14]. This IDE comes with a lot of handy features that allows for simpler programming. One can easily design how the user interface

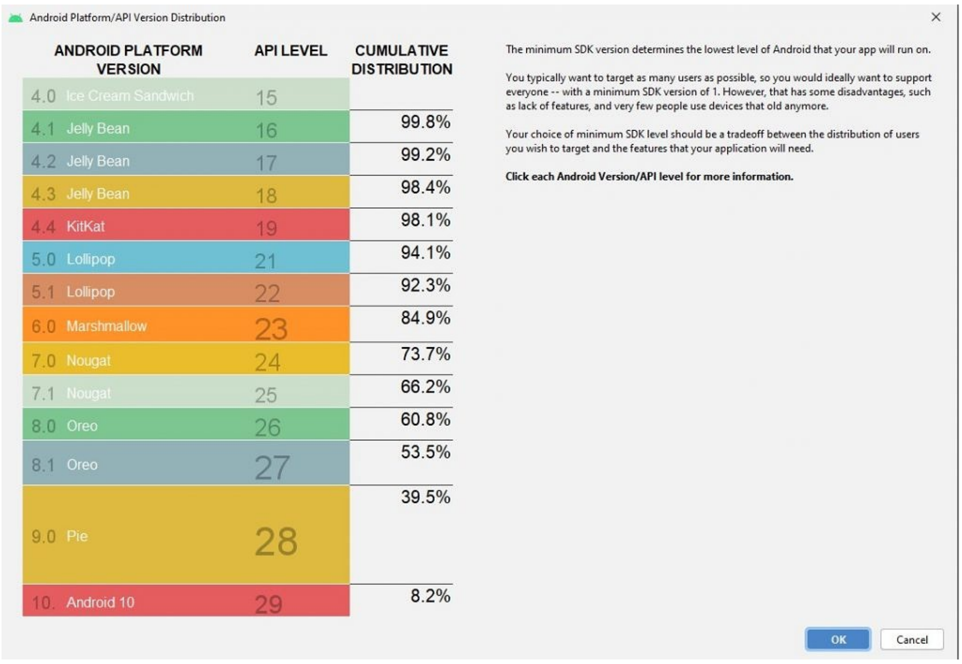


Figure 5.8: Android version distribution [27]

will behave and look like from their built in XML viewer and they also have the option to run the application on virtual android devices. Users can also build the application directly to their own phones, if the phone is connected to the computer via a USB connection and the phone has USB debugging enabled. Being able to test the application directly on the phone is highly useful as one can observe how well the application runs on a real life device, and also providing the ability to test it in the field.

5.4 Mapbox

Using mapbox studio turned out to be relatively simple. The user interface is quite simple to interact with, and they also provide excellent guides on how to implement the different features they have available. For this project a new style of map was created in their studio site (see fig 5.9). The outdoor template was selected as this was the most detailed option for outdoor environments, however as described in the research chapter this would also prove to not be detailed enough.

A tiling layer was then decided to be added to the map view via the style view interface (see fig 5.10), the tiling layer would be populated by GeoTIFF files provided by Kartverket.

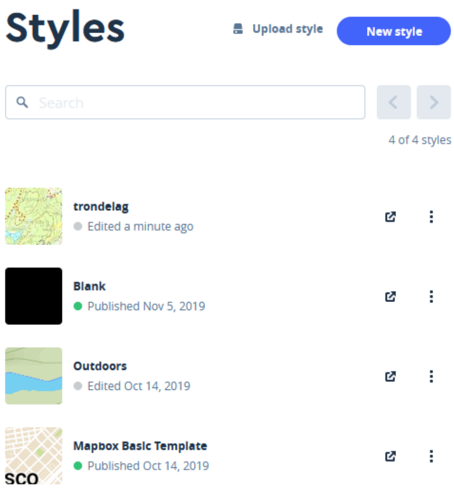


Figure 5.9: Mapbox Studio style overview

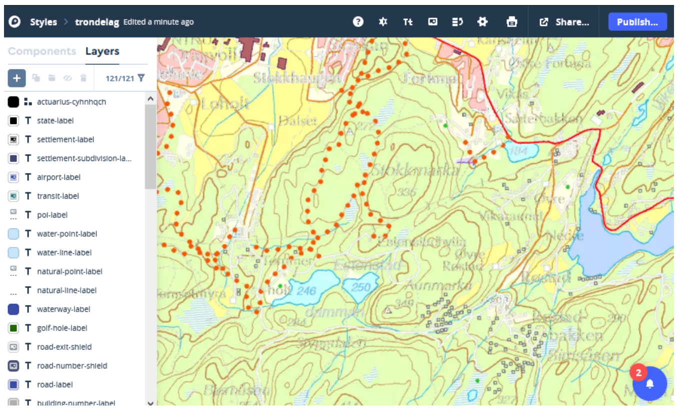


Figure 5.10: Mapbox Studio style detailed view

5.4.1 Kartverket

Kartverket provides maps for the whole of Norway upon request and in many different forms, but to get the level of detail that was required for this project the N50 raster was selected [12]. These are provided in 20 km x 20 km GeoTIFF files, which was tested on the mapbox map. While testing it was discovered that it would leave large overlaps of the maps, there was also problems with black borders around the map sections and also the amount of files one could upload to mapbox was limited. Thus a new solution had to be used and so attempts at stitching these GeoTIFF files together began.

5.4.2 GDAL

The tool of choice to add together the GeoTIFF files from Kartverkets download was GDAL, which is a translator library which is specifically designed for working with file types such as the GeoTIFF format [11]. The first attempt was to make one large map for the whole of Norway, but upon attempting to upload this to Mapbox, the upload failed as the file was much too large. Which led down a road of attempting to split it up into slightly smaller pieces that Mapbox could handle. Finally upon splitting the Norway sized map into 4 pieces and removing all files that included only water, it finally worked to upload it and apply it to the map. This means that the whole of Norway is not supported as of this version, which limits the applications overall usability as of now.

5.5 Google Firebase

This section will describe the different features of Google Firebase that was implemented in the application.

5.5.1 Authentication

The authentication for user login was done by Google Firebase. This is very simple to implement in the application with basically creating the landing activity (meaning the first "page" of the application that the user can see) to be the login activity which starts a call to the built in sign in intention provided by Google Firebase.

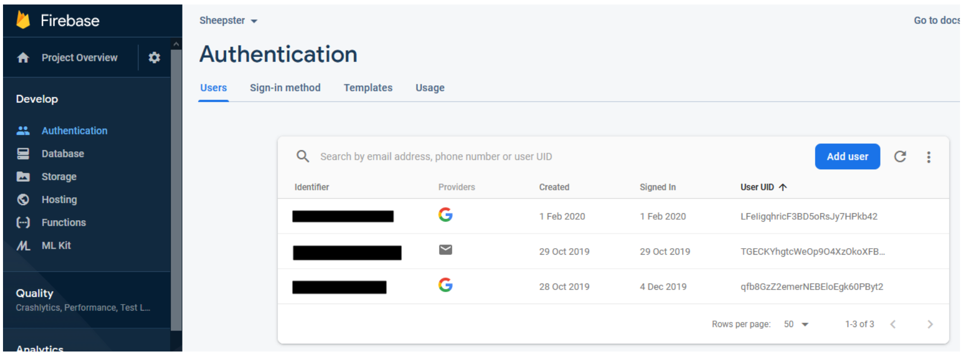


Figure 5.11: The Firebase authentication interface (email addresses removed for privacy reasons).

Whenever a new user has logged into the application, their email address is available to view in the Firebase authentication interface (fig 5.11). One can also add new users here manually which is ideal for testing purposes. By using Firebase authentication one automatically also gets to use their built in services such as the forgotten password functionality, which will send an email to the user that requests it with a link for resetting their password. Logging in as a new user is also very easy, the system automatically detects if you have been logged in before, and if you have not it will register you to the system. If the user is already logged into their Google account on the device in question, it can also use that to authenticate the user.

5.5.2 Database

As mentioned in the research section of this report, the database from Firebase is a NoSQL database which behaves as a storage of objects. Once the user has successfully logged in, the application will automatically be populated with the available data for that user from the local Firebase storage. The local storage is always available even when the mobile phone is offline, and so the application behaves as normal even in offline conditions. Once the device is online it will attempt to re-sync with the online database, including uploading any new data. Whenever new data is being made in the application, one basically just tells the Firebase database to be overwritten with the new objects that have been created.

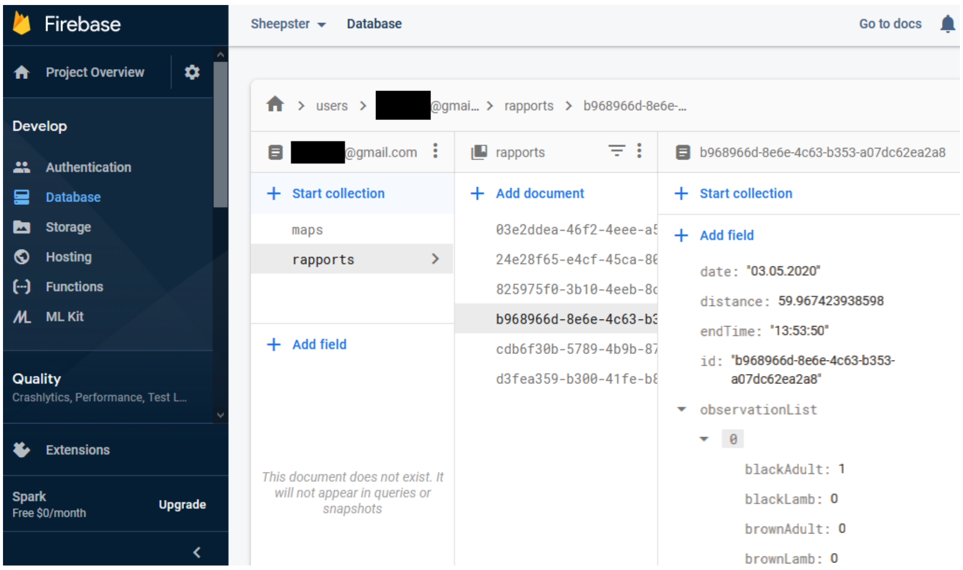


Figure 5.12: The Firebase database interface (email address removed for privacy reasons).

As seen in figure 5.12, two collections has been created for the user. This is ”maps” and ”rapports” (should be ”reports” but this error was not noticed until after the application was determined to be finished). The maps collection was intended to store the location and size of each offline map that the user has created and downloaded. Thus allowing it to be displayed on multiple devices, and would also make it possible to re-download all existing maps on any new device that the user acquires. The report section contains a list of all reports that are registered to the user. This is where all location data, all observations etc are being stored.

5.5.3 Storage

The storage functionality of Firebase was used to store snapshots of the map where the observation took place. Basically upon creating a new observation point, the application uses the location of the user and the observation to create a snapshot that contains both points. This is then uploaded to the Firebase Storage. Unlike the database this does not function very well offline. There is no local buffer for this feature at this point in time, and so any image functionality will not operate properly offline. Uploading images could work if the application is kept in running in the background until a data connection is available, as it will attempt to upload the image at set intervals. The uploaded images can be viewed in the Firebase console if one desires (as seen in fig 5.13).

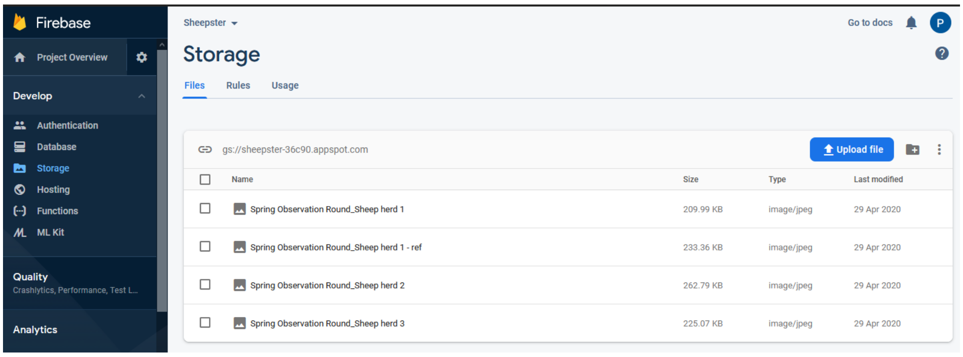


Figure 5.13: The Firebase storage interface.

This functionality of the application ended up being a lot less refined than the others, as it was one of the very last features to be worked on and it is not essential to the overall functionality of the application. While it does operate as intended with it creating a proper image for the observation, it does not have any indicator of where the observation was done, or where the user was standing. There was several attempts at fixing this by using different methods of taking an image of the map provided by mapbox, but no satisfying result was ever achieved. One could potentially draw on the image itself by using the size of the map, the known location of the map, and the known coordinates of the user and observation, but this was not looked further into.

5.6 Application

This section will describe how the final application looks and behaves like on a page by page basis.

5.6.1 Login

The login activity handles the authentication of a user's identity, and it achieves this by using Google Firebase's authentication functionality.

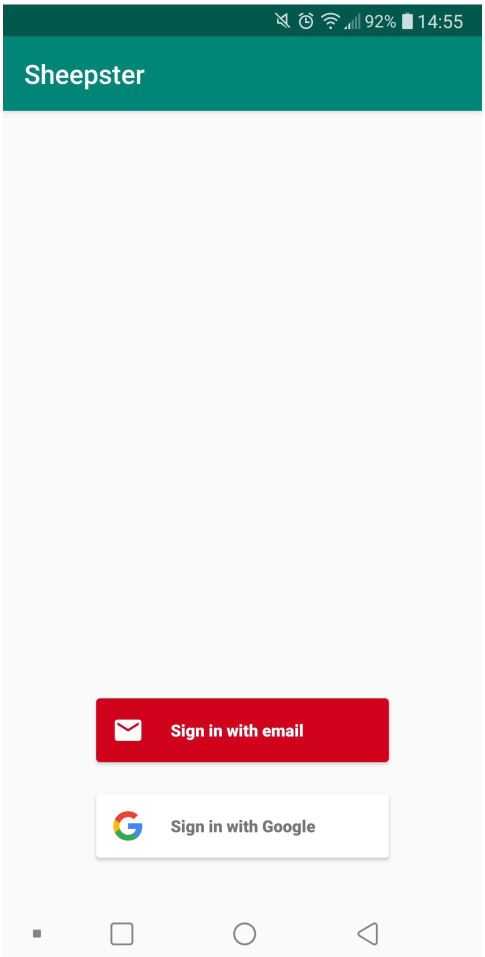


Figure 5.14: Login activity of the application.

Users can log in with either a google account, or with any other normal email address (fig 5.14), and so if the Android device already contains a valid Google log in, the user can import that to this application.

If an email address is provided that the application has not seen before, the user will be transferred automatically to a sign up page where the user inputs some information about them, and also sets a password.

5.6.2 Maps

The map activity of the application is used to store maps for offline usage, viewing stored maps, or deleting stored maps (see fig 5.15).

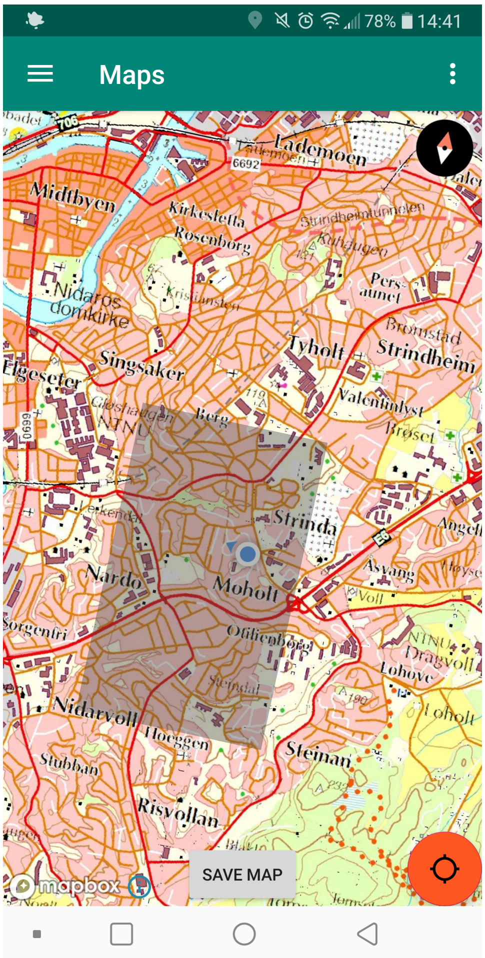


Figure 5.15: Map activity of the application.

In order to store a map for offline usage, the user clicks the save map button located at the bottom. When clicking this the user will be prompted asking them to name the new section of map, after a name has been put in and the user clicks download, the application will start saving that section of the map. After completion a prompt will let the user know if it was done successfully or not. The maps that are stored are displayed as grey rectangles, and when clicking them the user can see the name of the stored map. Doing a "long click" on the rectangle will prompt the user if they wish to delete the stored map. Accepting this prompt will delete the map. The button on the bottom right will centre the map over the user, and the button in the top right will only appear if the map has been rotated and clicking it will realign the map properly with North being the top and South being the bottom (see fig 5.15).

5.6.3 Reports

The reports activity is a list of the reports that the user has created in the past (see fig 5.16).

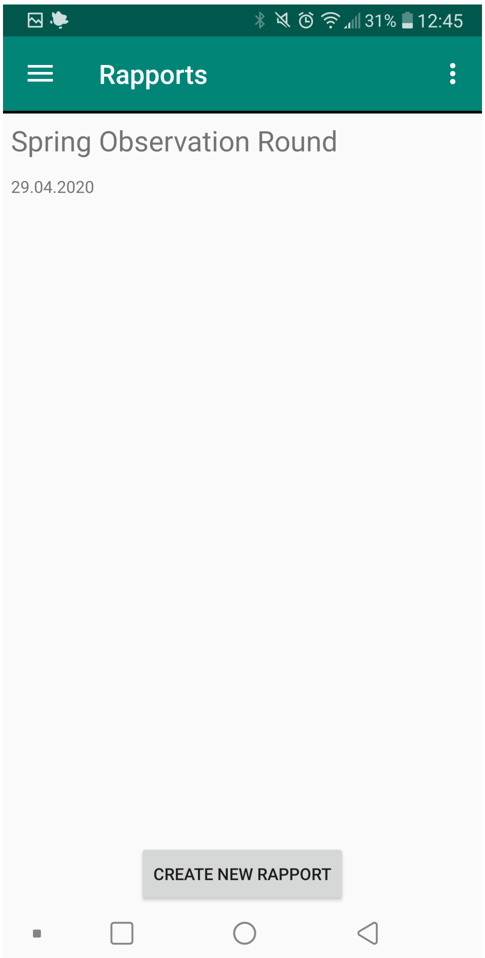


Figure 5.16: List of all reports for this user.

This is also where a user can create a new report by clicking the ”Create new rapport” button at the bottom. Clicking the button will give the user a prompt asking them to name the new report, and upon completion a new report will be added to the list. Each report is clickable and will navigate you to the report overview page.

5.6.3.1 Report Overview

In the report overview, the user can view a lot of relevant data for the current report. Such as the name of the report, when the report was started and ended, the total time elapsed on the trip, and also the total distance for the whole trip. The user can also set the weather condition for the trip using the drop-down menu (see fig 5.17).

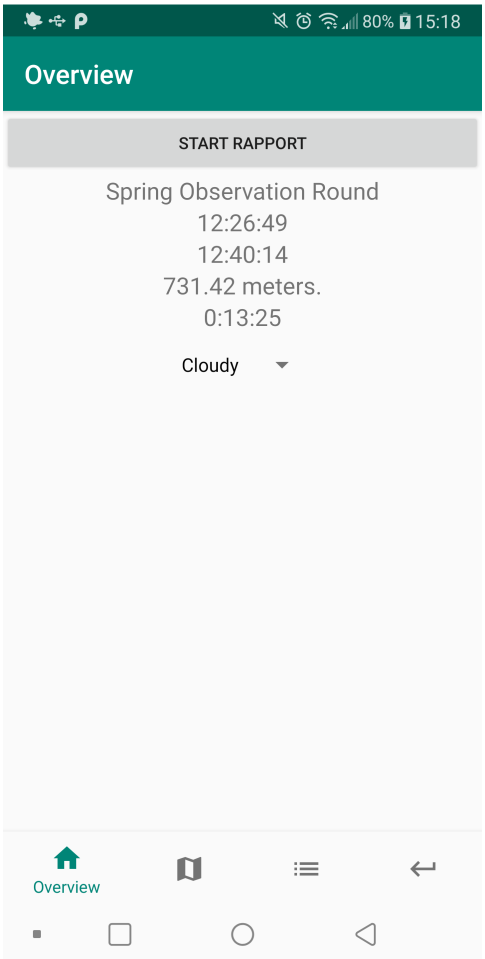


Figure 5.17: Overview of the current reports in the application.

By clicking the start button, the application will start to record the user's position, and also update the distance travelled by calculating the distance from the new position compared to the previous location. The button will also then change to be a stop report button, which when clicked will fill the stop time section of the report, and also stop recording user positions. The amount of time spent in total will then also be displayed right below the distance.

5.6.3.2 Report Map

In the map view, the user can see their current position on the map, along with where they have travelled, where any observations have been made and where they have been made from (see fig 5.18). If the report is currently set to be running, the application will continue to draw dots of where the user is located every 5000 milliseconds. Clicking anywhere on the map will centre the map on the click, and using the button in the bottom right corner will centre the map over the user. Users can also click the observations to enter them and display/edit the data within them.

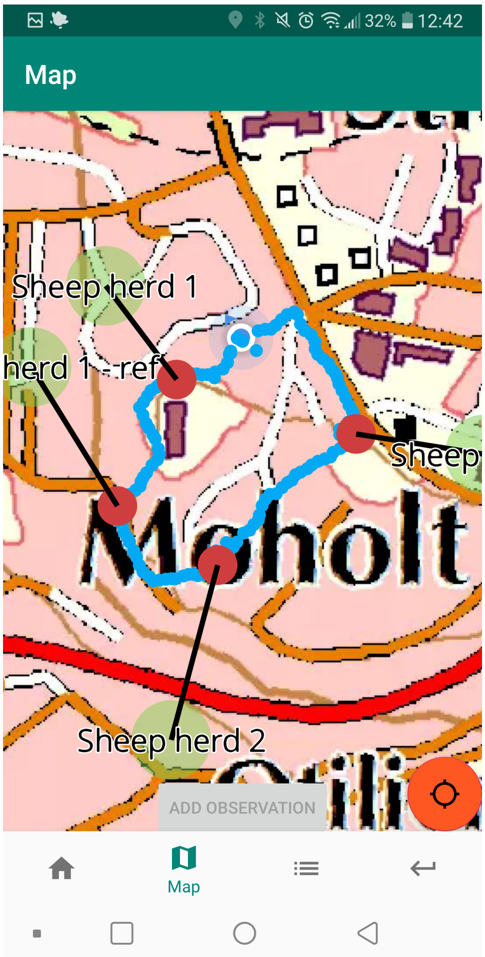


Figure 5.18: The map view of the current report.

If the report is currently running, the button on the bottom of the view is active and users can add a new observation. This will use the middle of the viewed map as the centre of the new observation, and prompt the user for both a name of the new observation and also what type of observation is being done. Upon a successful input, an observation will be added to the map.

5.6.3.3 Report Observation List

The observation list view is a list of the observations that are registered in the application (fig 5.19). The user cannot register new observations from here, but they can click any of the observations to enter the detailed view for each of them. The list view tells the title of the observation, what type of observation it is, and also when the observation took place.

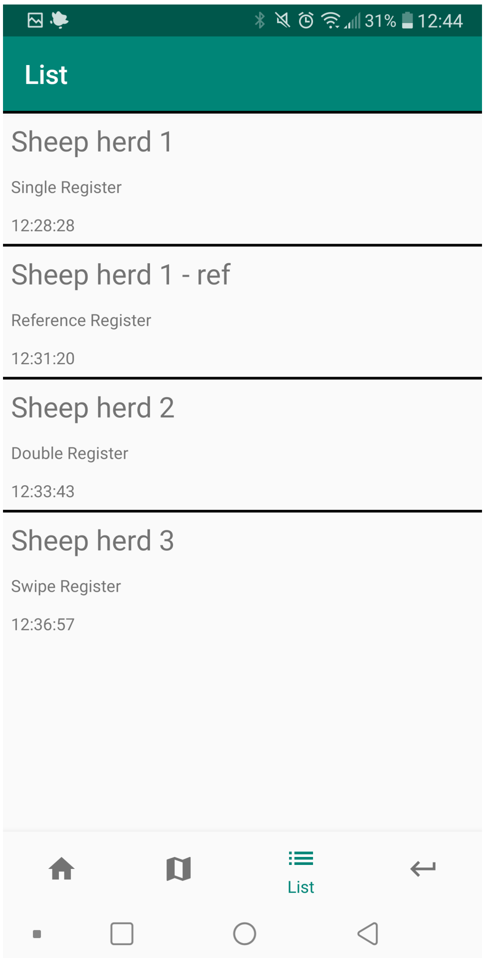


Figure 5.19: The observation list view of the current report.

5.6.4 Observation

In the final version of the application, one would preferably only have one type of main method to register the sheep. In this version however, there's currently three different methods of registering sheep, and a fourth type to register a reference to previous flocks of sheep. For this prototype all three methods have been kept in so that testing could be performed on each of the different methods and thus determine which method is the preferred way in both usability and user experience.

5.6.4.1 Observation Overview

The overview of each method of registration is much the same, only with some different methods of interaction (see figures 5.20. 5.21, 5.22, 5.23). Common for all the observations are that they contain the title of the observation on the top of the view, followed then by the time the observation took place. It also displays the position of the user at the time of registration and the position of the observation. By using this data it calculates the distance between the user and the observation automatically. Each type of observation also has views for displaying how many sheep they currently have recorded. The snapshot discussed in chapter 5.5.3 is also displayed at the bottom of this overview.

5.6.4.2 Single Registration

The first method of registering sheep that was implemented in the application was the so called "Single register". When the user interacts with the register sheep button, the user is then navigated to the registration view (fig 5.20).

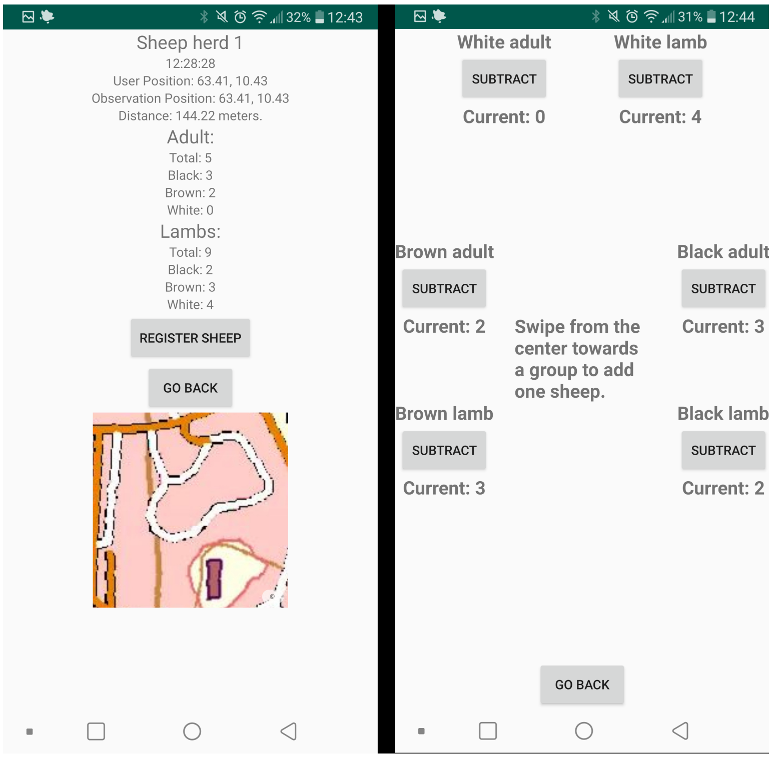


Figure 5.20: The single register fragment of the Sheepter application.

By swiping from the centre towards any of the sheep names on the edges of the screen, the amount of sheep associated with that direction will be increased by one. In order to remove any sheep (in case of a wrong registration), the user would have to look down on the screen in order to click the subtract buttons. This plus the fact that each type of sheep was so close together, made this type of interface difficult to interact with and so further development had to be done.

5.6.4.3 Double Registration

The second method of registration that was developed was the ”double registration”. In this iteration the sheep have been split into two groups, being the adults and the lambs (fig 5.21.

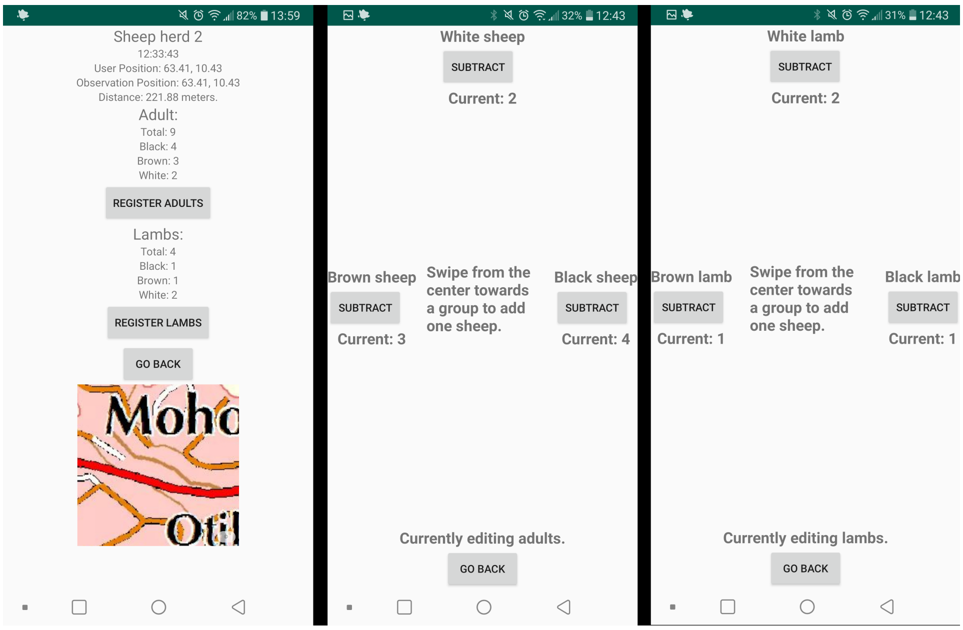


Figure 5.21: The double register fragment of the Sheepter application.

The method of functionality here is much the same as the single registration, with the user swiping from the centre towards the sheep they wish to add, and with the added space of removing half of the sheep it is now easier to register the correct sheep. However the issue with subtracting sheep here remains the same as the previous version. An argument can be made that it is slightly simpler to learn where the buttons are now as there are less of them, but it was clear that a new version would have to be designed.

5.6.4.4 Swipe Register

The final method that was created is the "swipe register". In this solution the user is currently only editing one type of sheep at a time (fig 5.22).

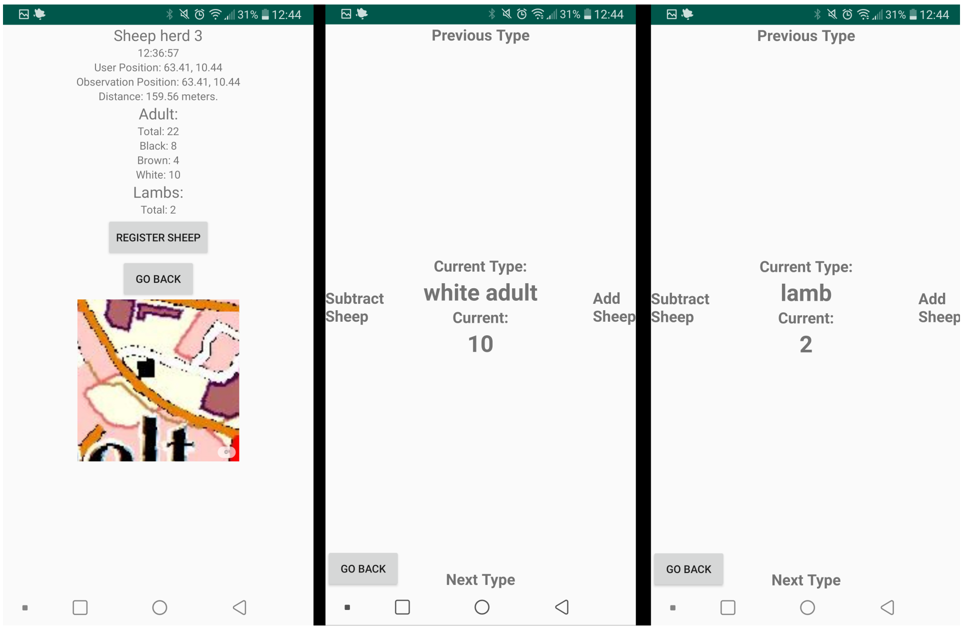


Figure 5.22: The swipe register fragment of the Sheepter application.

The user here can subtract or add sheep by swiping from the centre of the screen to the left or to the right respectively. In order to switch between the different type of sheep, the user will swipe up or down. This version also has a simplified method of registering lambs by only registering them as a group and not based on their colour. This was done as the method for differentiating between the detailed registration and simplified registration was not implemented, and so this provides as an example of how the simplified registration of lambs could work.

5.6.4.5 Reference Register

The reference register method is a registration of a previously observed flock. As such it has no registration button, but rather a drop-down menu that is populated by the observations that had been done previously (see fig 5.23). All future events that happened after the reference register is greyed out to prevent a user from referring a flock of sheep they saw later in the tour.

In order to edit the sheep of that herd one has to edit the original herd.

The reference register has been added so that the user can see where the herd has moved since the last time they were observed.

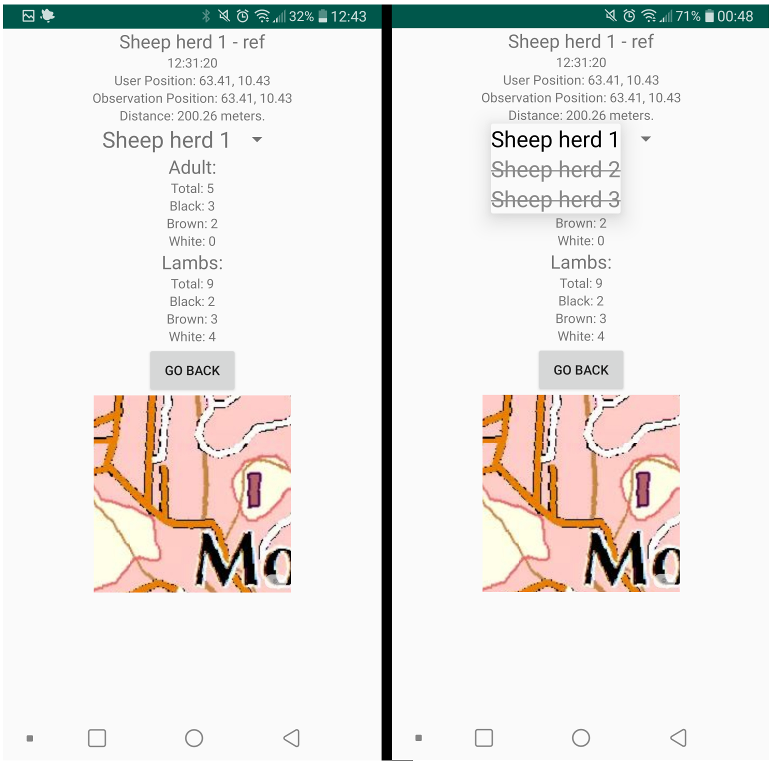


Figure 5.23: The reference register fragment of the Sheepter application.

5.7 Text-To-Speech - TTS

One of the major requirements of the application is the ability for the sheep farmer to register sheep on the mobile device without having to look down from their binoculars. There was several methods discussed on how to achieve this, should the device vibrate or flash a colour when a sheep is registered or have some form of vocal output from the device. The latter turned out to the most successful method.

In the first iterations the application would merely play a sound file telling the user what the colour of the sheep that was registered is, and this was achieved by using prerecorded audio snips. Later however after a lot of research and testing, text to speech was discovered. This made a lot of sense to use, as the application could then be talking to the user directly, and telling them what they are doing without the user having to look down. So in the latest iterations, the device uses the built in text to speech functionality of Android, to tell the user how many sheep they have recorded of a given type whenever they register a new sheep. In the swipe register it will also tell the user what group they swiped to, and also upon double clicking the page it will re-iterate what group they are currently editing.

Chapter 6

Testing method and execution

6.1 Method

This section of the report will go into what method of testing was selected for this thesis, and also why it was selected. It will also describe how the different tests were produced, and how they would be performed.

6.1.1 Focus Group

Figure 6.1: Focus group.

As mentioned in the introduction, the testing of the application will be completed using focus groups. Focus group testing is ideal for when there's a new product that needs to be tested before being released to the public at large. The author also has previous experience performing focus group testing from his bachelor degree and so a lot of the same methodology could be used for this project.

To perform a focus group test, one takes a diverse group of people and ask them to interact with the new product. Meanwhile the researchers will collect data both by observing how users use the product and the results they achieve with it, and also by how the users themselves rate the experience with the product.

Usually when one does this in software development, it is customary to have a preliminary meeting with the focus group to determine what features they are expecting out of the product before producing the software. However for this thesis the requirements of the application have already been determined by the contractor and so this part of the methodology has been cut.

The testing of the application will be three folds, first a usability test will be performed with minimal guidance from the author, afterwards an anonymous questionnaire will be filled out by the tester to determine what their user experience was with the application. Lastly an interview will be conducted with each test subject. These methods will be explained in further detail in later sections.

6.1.1.1 History and theory

Focus group testing has a rather macabre origination story. The first tests ever done was performed by the US military to determine the impact of military propaganda films on the viewers [37]. In 1944 the previously named Office of Radio Research had moved to Columbia and changed their name to the Bureau of Applied Social Research. With this move they had also decided to change their focus from media commentary to influential social research. In this time a sociologist named Robert K. Merton was added to the team. Merton is said to be the so called ”father of the focus group”.

It was shortly after Merton had joined the ranks that the propaganda testing begun. This is when they developed the focus group interview, and in their own words it was going to be used to ”elicit the responses of groups to text, radio programs and films”. The methodology in later years had continued to develop and improve in many aspects and today is a wide spread method for testing products.

6.1.1.2 Target audience

The target audience for this application is diverse in some aspects and uniform in others. It is hard to determine a specific age group as sheep farmers come at all ages. While mobile phones and applications in general are more used by younger generations, it would still be very valuable to reach as many age groups as possible while testing. Participants that work in the field of sheep farming would invaluable testing subjects as these are the actual clients that will potentially use the application once it has reached a finished state.

6.1.1.3 The test subjects

The test subjects for this project are not as varied as was originally planned. This is because of the current health situation in Norway and the self quarantine measures that have been taken. So the individuals that the application have been tested with are all living in the same floor as the author, and the age range is not very wide. None of the testers are sheep farmers nor have they had any experience with this in the past and so they may not completely understand why such an application would be required. Both of the testers have studied IT at some point and are now working full time with it. This will naturally also affect the results provided from the testing as they have experience with developing software on a regular basis.

One of the subjects however is visually impaired, meaning he has reduced vision and has to wear glasses to properly see. This will help with providing a different perspective of the usability.

The testers are as follows:

- Subject 1: male, 25, manufacturing engineer.
- Subject 2: male, 25, software developer with a masters in Computer Engineering.

Preferably the situation would be quite different and proper testing should be performed with further iterations of the application before any attempts of release to the public.

It is also important to note that while the tests are performed mostly anonymous, there will still most likely be bias towards higher scores as a cause of the subjects being personal friends with the author.

6.1.2 Usability

In this usability chapter, the thesis will describe the history of usability testing, and how it will be performed for the application that has been developed.

6.1.2.1 History and theory

Usability is a relatively new term. Before 1990 the same family of metrics was named ”user friendliness” and it took a long time before researchers in the field could agree on a set definition for what exactly makes a product ”usable”.

Usability refers to the ease of access and/or use of a product. It is a part of the overall User Experience methodology and was previously used interchangeably with this label, but it has in later years been defined as only one part of the whole picture that is User Experience. Unlike the experience of using a product which cannot be directly measured, usability can, and so one performs usually some form of interactive interview.

The usability interviews, or tests if you will, are structured one-on-one interviews with focus on the product at hand. The test is done by having the user partake in a set of tasks that have been preliminary determined by the researcher/interviewer. During the testing the researcher will continuously be making notes on what the participant is doing, how they are performing, and how they experience the product. After the tests have been run on all test participants, the results can be compared and used to determine both what is good about the product and what issues it may be facing in its current iteration.

6.1.2.2 Conducting the tests

In order to properly determine the usability level of a given product, the researcher has to inspect the feature of the product with the user, they have to determine what the user wants to do with the product and also what the environment will be for performing the tasks. The environment for this thesis' application would naturally be out in the field with the animals.

How usable an interface is can be determine by the three main factors that are displayed in figure 6.2, and are also described below.

1. **Feel:** How easy the user becomes familiar with and competent in using the user interface during their first interaction with it.

2. **Usability:** How easy it is for the user to achieve a given objective.

3. **Look:** How easy it is for the user to learn the user interface and thus know how to use it on subsequent tasks.

Figure 6.2: Usability factors.

The above factors is what will be primarily focused on during the testing of the application. After the tests have been executed, the participants will work with a questionnaire that attempts to score their experience with the application. It is at this point that the testing goes from usability to user experience.

6.1.3 User Experience

This section of the report will describe the methodology of User Experience, the history of it and how it will be used for this thesis.

6.1.3.1 History and theory

User experience has its roots way back in the ancient science of ergonomics, which was an attempt at establishing a set of principles that would make work more convenient and efficient. The first encounter with this was from Hippocrates description of what an optimal workplace would be for a surgeon [24].

In the 1900s Winslow Taylor spearheaded the more modern optimisation of work, focusing his research in the interaction between a worker and their tools. Later in 1940 Toyota followed with their so called ”Human-Centered-Production system”, which was aimed at improving the efficiency at their facilities by modifying the environment for the workers [36]. The product side of the modern user experience design also saw great progress in the early 1940’s with Alan Turing’s first theoretical computer. Without this the field of computer science would not have developed to the stage it is at today. In 1955, Hendry Dreyfuss wrote the famous book ”Designing of People”, which is seen by many as one of the major milestones for the birth of User Experience Design. In this work he described the importance of the connection between people and their experience for the success of a product. Later in the year 1995, the famous cognitive psychologist and designer Don Norman coined the term we know so well today, ”User Experience”, to describe the methodology that his team was using for their work at Apple Computers. From here it further spread throughout the computer science society and is now today a major cornerstone in any successful product.

The Nielsen Norman group defines user experience (UX) as follows: ’”User Experience” encompasses all aspects of the end-user’s interaction with the company, its services and its products’. This essentially means that UX is about how a user feels while interacting with the product and is highly subjective. Because of this subjective nature it is difficult to get any form of real measurement, but one can attempt to generalise it.

There are four major elements to UX, these are namely: usability, value, adoptability, and desirability [6].

1. **Usability:** How simple is it to complete a given task?

2. **Value:** Does the product have a value?

3. **Adoptability:** Would people use the product?

4. **Desirability:** Is the experience of interacting with the product fun and engaging?

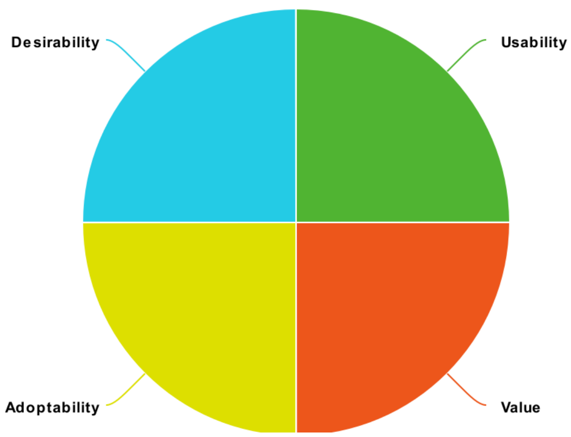


Figure 6.3: The four elements of user experience [6] .

6.2 Execution

In the following list, the report will go over how the execution of these tests will be performed.

1. Usability testing of application.

    - The usability test will be performed individually and as separated from other people as possible in order to prevent any form of outside contamination of the experience and result of each participant.
    - The participant shall first be introduced to the application and given a short period of time to navigate it and get to know it.
    - The participant shall then perform a set of tasks, which is as follows:
        (a) Save a map locally to the device.
        (b) Create a new report.
        (c) Start the new report and change the weather conditions.
        (d) Register three new observations using the different methods of registration (as listed below), and populate these with a predetermined amount of sheep.
            i. Single registration
            ii. Double registration
            iii. Swipe Registration
        (e) Lastly, the tester will be informed to create a new observation that is a reference to a previous observation.
    - The amount of time that the users spend on performing the tasks, as well as any failures/successes will be recorded by the supervisor of the test.

2. UX testing

    - After all usability tests have been performed, each participant will answer a survey regarding their experience with the application. The survey is composed of two parts, the first being about the overall experience that the user had with the application, and the second part is about each specific method of sheep registration. The statements regarding the registration will be the same for all types, in order to get an understating of how the different methods compare to one another.

3. Interview

    - After the testing and the survey has been completed, an interview with the test subject will be conducted. This is to give the test subject an opportunity to give direct feedback to the author regarding the application in its current state, in addition to possible suggestions they feel would make the application better.

Chapter 7

Results

This following chapter will contain the results from the focus group testing that was described in the previous pages. This includes the creation of the focus group, the usability test, the user experience test and the interview with the test subjects.

7.1 Focus Group

Creating the focus group was relatively easy in this case as there was not a lot of individuals to chose from, given the formerly mentioned situation. As both testers have full time jobs, it was decided that the testing would be taking place on a Sunday.

7.2 Usability Test

In this section the report goes into how the usability tests were performed, and what the observed results were.

7.2.1 Preparations

To prepare for the usability test, a rudimentary test was designed. The test had to make use of each feature that the application has to offer, in order to give the user a full understanding of what the application is created for and meant to do. This will affect their overall experience with the application and ideally provide a solid basis for them to rate their user experience.

7.2.2 The tasks

The first action that each user had to perform was to save a map locally to the device, and then delete it afterwards. After this they each created their own report to test with, and selected the weather conditions that most suited the weather outside.

In order to make sure the test subjects knew how to use the map properly to register sheep observations, three real life locations was provided in the instructions. Alongside with these locations were the numbers of sheep they were expected to register at each location. The method of registration was also provided. Lastly they each had to register a new observation of a previously observed flock, the location of this group was not determined in advance and the subject got to choose the location on their own.

7.2.3 Results

7.2.3.1 Map saving

While both users accomplished the task of both saving a map and deleting it, it was clear that they had issues with the user interface. It was not really clear what the boxes on the map was supposed to indicate, and so this had to be told to them in order for them to understand it. It was also not directly clear how to delete a map. As a single press only shows the title and one has to do a long press to open the delete menu.

7.2.3.2 Starting a new report

Creating a new report was an easy task for both subjects and was completed in no time at all. The selection of the weather conditions was also done quite rapidly without any guidance.

7.2.3.3 Single Registration

The registration of the location on the map was done with minimum effort by the subjects, subject one commented that the map looked a little bit weird in comparison with how Google maps appears.

During the registration each subject was informed to register 6 white sheep, 7 black sheep, 6 brown sheep, 2 white lambs, 4 black lambs and 1 brown lamb. Subject one performed excellently here and did not have to look back down at the mobile device after the first viewing of how the interface was laid out. Subject two however had to look down multiple times in order to remember where each type of sheep was located.

Neither subject made any mistakes during their registration. However, they were told to remove one sheep from the white adults section, which made both users have to look down at the mobile device.

7.2.3.4 Double Registration

The location registration on this registration was also done easily by both subjects and took very little time.

This time each subject was instructed to register 3 white sheep, 8 black sheep, 1 brown sheep, 7 white lambs, 4 black lambs and 2 brown lambs. This time both subject performed quite similarly in a positive manner. Neither subject had to look back down on the screen during registration and subject two commented that it was a lot easier this time around as each direction corresponded to each type of sheep.

No mistakes were made here either, but the same result occurred when users were informed to remove one sheep from the brown adults section.

7.2.3.5 Swipe Registration

Location registration went quite well for both subjects on this test as well.

For this test each subject had instructions to register 5 white sheep, 9 black sheep, 2 brown sheep and 6 lambs (as this method has a simplified lamb registration).

Both subjects performed excellent on this test with no errors. Subject one commented that this method was a lot easier to perform as the application told him exactly what type was currently being edited.

When told to remove one sheep from the black adults section, both users could perform the task without having to look down at the screen.

7.2.3.6 Reference Registration

Lastly each subject was told to create a reference registration to give them a full view of how the application works. The location of this observation was left up to each user, before they were then instructed to make it refer to the single registration group they had created at the start of the tests. Both subjects managed to perform this task without any errors.

7.3 User Experience Test

After the usability testing had been performed, it was time for the user experience test. This section will go into exactly how that went, in addition to how the preparations for the test was executed.

7.3.1 Preparations

The environment which the user experience questionnaire was chosen to be developed in was Google Forms. This allows for simple forms with predefined questions and input fields.

For this testing, a series of statements about how it felt using the application was created, alongside a scale that ranked from 1 to 7. A value of 1 would mean that the subject strongly disagrees with the statement, and a value of 7 means that they strongly agree.

The setup of the questionnaire is loosely based upon examples provided by Gary Perlman on his webpage [26], which can be seen in the introduction section of the webpage.

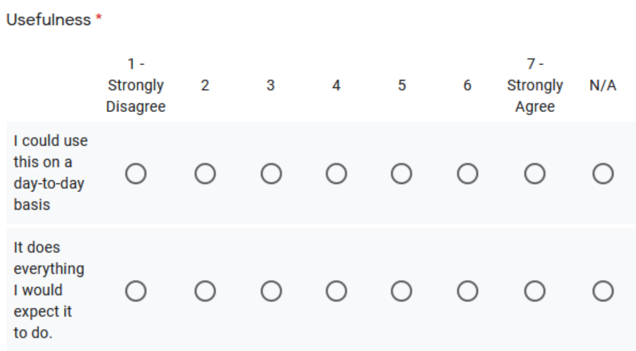
Each user was also informed that they should answer this as if the application would be relevant to their line of work, or rather put themselves in the shoes of a sheep farmer that currently does this with pen and paper.

7.3.2 The tasks

The form was divided into 4 major parts, where each part would correspond to a different aspect of the application. The first part makes a series of statements about the application as whole, while the last three are specifically about each method of registration of sheep. This was added in order to allow the subjects to directly score how it felt using the different methods, and thus provide data that can be used for further development.

First they rated the usefulness of the application. This section is meant to extract how useful the application would seem to be in a real life situation for the subjects (see fig 7.1).

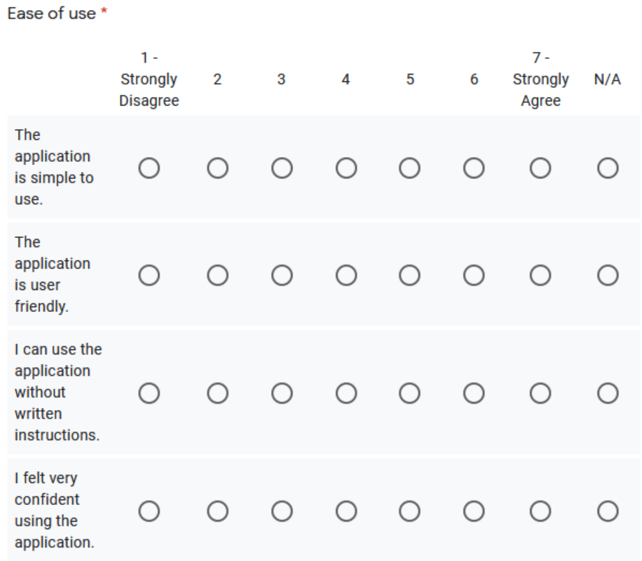
7.3.2.1 Overall impressions


Usefulness *

	1 - Strongly Disagree	2	3	4	5	6	7 - Strongly Agree	N/A
I could use this on a day-to-day basis	○	○	○	○	○	○	○	○
It does everything I would expect it to do.	○	○	○	○	○	○	○	○

Figure 7.1: The usefulness section of the questionnaire.

The ease of use section is next, and in this the subjects are given statements regarding how simple it was to use the application in general, in addition to how confident they felt while using it (displayed in fig 7.2).



Ease of use *

	1 - Strongly Disagree	2	3	4	5	6	7 - Strongly Agree	N/A
The application is simple to use.	○	○	○	○	○	○	○	○
The application is user friendly.	○	○	○	○	○	○	○	○
I can use the application without written instructions.	○	○	○	○	○	○	○	○
I felt very confident using the application.	○	○	○	○	○	○	○	○

Figure 7.2: The ease of use section of the questionnaire.

The section for ease of learning allows the subject to grade how easy they felt that it was to learn how the application works, how user friendly it was and how fast they felt they mastered it (figure 7.3).

Ease of learning *

	1 - Strongly Disagree	2	3	4	5	6	7 - Strongly Agree	N/A
I learned how to use the application quickly.	○	○	○	○	○	○	○	○
It is easy to learn how to use the application.	○	○	○	○	○	○	○	○
I quickly became skillful with the application.	○	○	○	○	○	○	○	○

Figure 7.3: The ease of learning section of the questionnaire.

The last set of statements are regarding the overall satisfaction the user had with the application. Meaning how fun it was to use the application, how comfortable they were using it, and whether the application worked the way they wanted it to work (see figure 7.4).

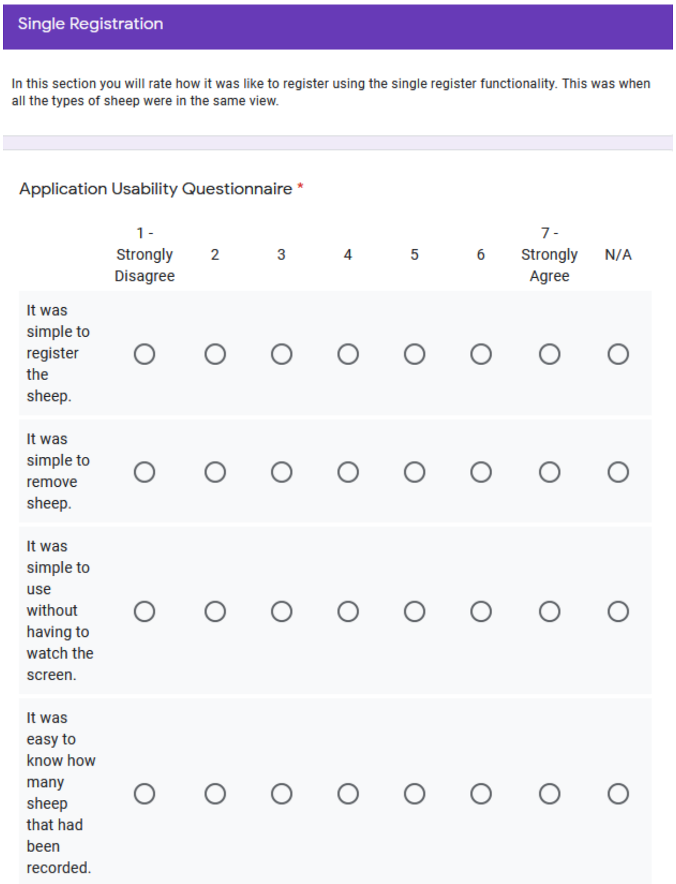
Satisfaction *

	1 - Strongly Disagree	2	3	4	5	6	7 - Strongly Agree	N/A
The application is fun to use.	○	○	○	○	○	○	○	○
The application is comfortable to use.	○	○	○	○	○	○	○	○
The application works the way I want it to work.	○	○	○	○	○	○	○	○

Figure 7.4: The satisfaction section of the questionnaire.

7.3.2.2 Sheep Registration

In the last three parts, the subjects were provided with the same set of statements (fig 7.5), with each corresponding to a method of registration. The series of statements for this part is about how it felt like using each functionality that the method of registration had to offer.



Single Registration

In this section you will rate how it was like to register using the single register functionality. This was when all the types of sheep were in the same view.

Application Usability Questionnaire *

	1 - Strongly Disagree	2	3	4	5	6	7 - Strongly Agree	N/A
It was simple to register the sheep.	○	○	○	○	○	○	○	○
It was simple to remove sheep.	○	○	○	○	○	○	○	○
It was simple to use without having to watch the screen.	○	○	○	○	○	○	○	○
It was easy to know how many sheep that had been recorded.	○	○	○	○	○	○	○	○

Figure 7.5: The single registration section of the questionnaire.

7.3.3 Results

7.3.3.1 Overall impressions

As can be seen in figure 7.6, both users gave the application a very high score. This means they they both felt the application performed as expected and it fulfilled all the work criteria that they had been explained to be required. This could also potentially just be bias towards the author, not wanting to give a friend a low score on something they created. It could also mean that the author had biases in his explanation of the work requirements thus providing the users a faulty starting out point.

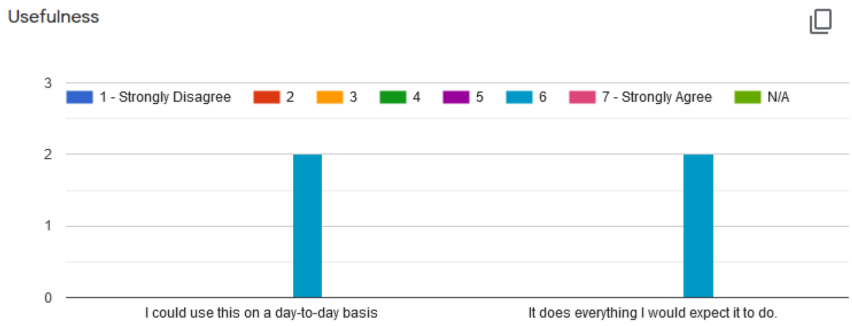


Figure 7.6: The overall usefulness result from the questionnaire.

For the overall ease of use result (see fig 7.7), there was some differences between the subjects. Overall they felt the application was decent in terms of user friendliness, the simpleness to use, and confidence while using it. However one of the subjects did not feel like they could really use the application without some form of instructions.

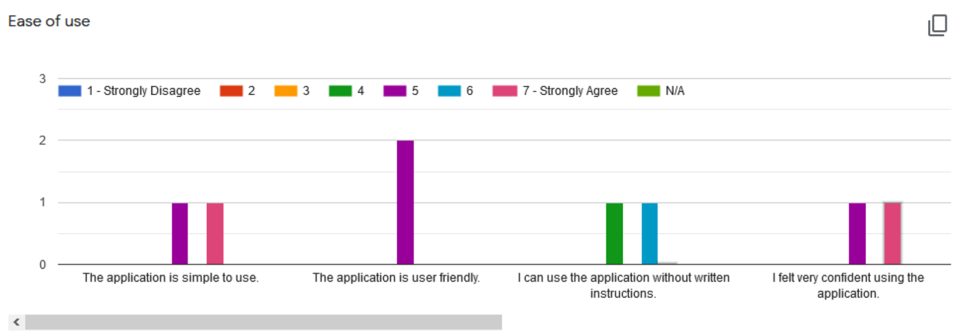


Figure 7.7: The overall ease of use result from the questionnaire.

In the overall ease of learning section (fig 7.8), the subjects agreed on most points. They felt they easily and rapidly learned how to use the application and also that they became skilful with it in the short time they played around with it.

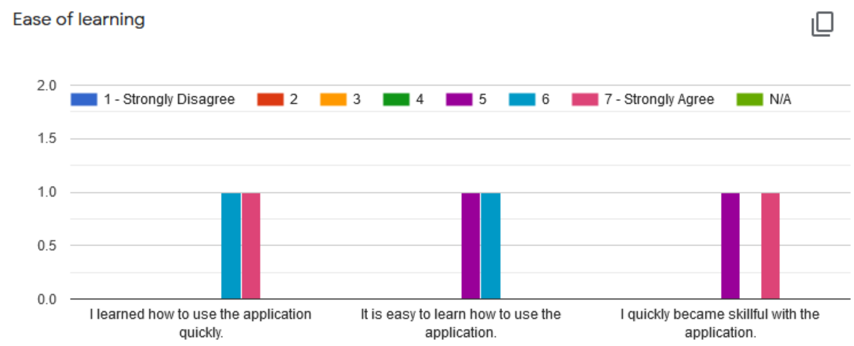


Figure 7.8: The overall ease of learning result from the questionnaire.

In the section for the satisfaction, the subjects had some disagreements (fig 7.9). One subject really liked using the application and found it to be fun, while the other was more on a neutral stand. However they both felt that the application was comfortable to use, and that it worked the way they wanted it to work.

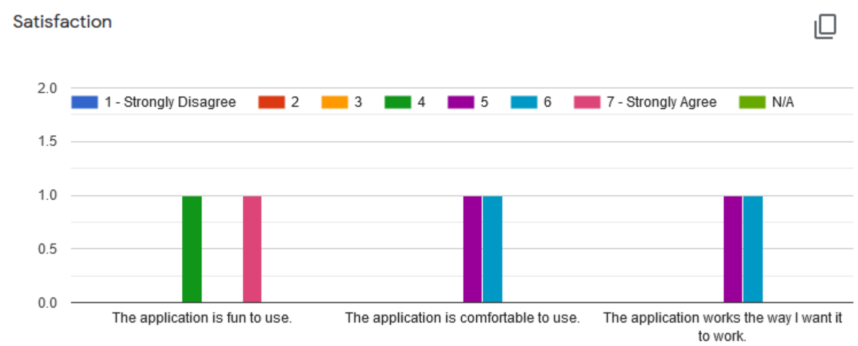


Figure 7.9: The overall satisfaction result from the questionnaire.

7.3.3.2 Single Registration

In the part of single registration it became clear that the major problem was removing sheep (fig 7.10). Other than that the subjects mostly found the method of registration to be quite simple. There was some differences in how easy they felt it was to use the application without watching the screen, but nothing major.

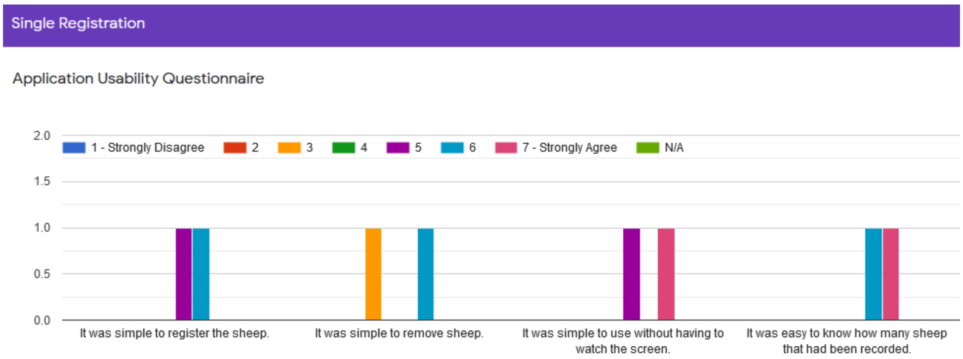


Figure 7.10: The single registration result from the questionnaire.

7.3.3.3 Double Registration

For the double registration method, the results was much the same as the single registration. However it was higher in general on all the scores except the removal of sheep (fig 7.11). Also the user that had issues with registration of sheep in the single registration now found it a lot easier.

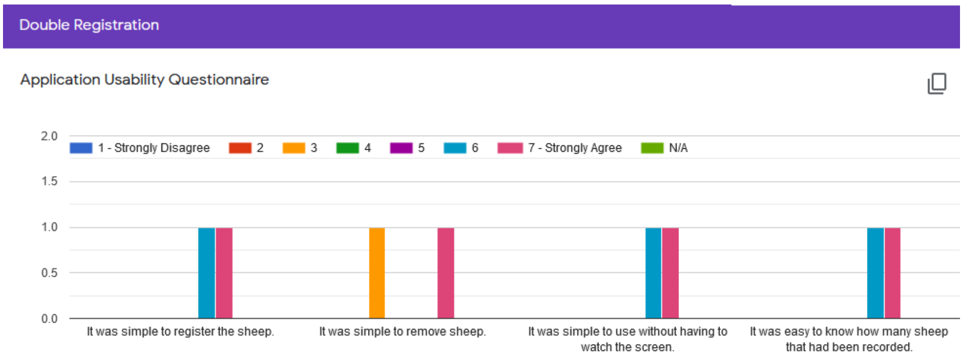


Figure 7.11: The double registration result from the questionnaire.

7.3.3.4 Swipe Registration

One of the users scored the method of swipe registration of sheep to be slightly lower than for the double registration (fig 7.12), but the removal of sheep was miles above what the other methods were scored. Both users gave it a perfect score. They also felt it was easy to use it without having to watch the screen, and it was also easy to know how many sheep had been registered.

Figure 7.12: The swipe registration result from the questionnaire.

7.4 Interview

After the usability and user experience testing, each subject was interviewed briefly in order for them to provide direct feedback with their own words. No real preparation was done for this part as this was more of a feedback session rather than a direct interview.

7.4.1 Subject 1

Subject one felt that the download bar was very hard to spot, and had not in fact seen it until the map was almost completely downloaded, thus causing him to question whether or not the download had actually started. The subject also felt that the input boxes for naming different elements should have a hint/example in them already, so that the user would know better what each input box will be used for.

He also stated that he preferred the swipe method the best overall, but felt like there should be a method of quickly changing to a specific colour of sheep without having to scroll through all other types.

7.4.2 Subject 2

Subject two had a lot of input in regards to the application and so this will be split unto further sub sections.

7.4.2.1 Registration

Overall the subject felt that the swiping towards a sheep to register worked quite well on all the different registration methods. He felt that the double registration felt a lot less cluttered than the single registration but that the single registration would probably be faster once a user gets used to it.

The subject appeared to prefer the double registration method as it appears he slightly misunderstood the functionality of the swipe method. This means that the interface has some shortcomings as it does not explain itself properly to every user, as it should.

7.4.2.2 Map

The subject felt that the map was easy to use and intuitive as it shares a lot of the same functionality as other map applications, but he felt that the quality was not very good. This point is slightly irrelevant as the map has been specifically selected from kartverket to be an accurate map of the outdoor landscapes in Norway, and thus it is not the best for urban usage.

The subject felt that adding an indicator on the view of where a new observation would be created upon hitting the add observation button would be valuable. This was actually intended in the beginning but somehow got lost during development.

7.4.2.3 General

The subject feels overall that the application feels a bit primitive and could use a lot of work on the user interface. Specifically a lot of the text is quite small and hard to read for visually impaired individuals and thus this should be made larger. The application also in general does not use all of the screen space that it has available.

The terminology that is used in the application could also see a lot of of improvement as it can sometimes be quite vague and confusing.

The subject also felt that some method of getting help within the application would be a good addition.

7.4.3 Summary

Overall both subjects agree that the overall user experience is pretty good. The registration of sheep was straight forward and easy to learn. Mostly the feedback they have is regarding the design of the application, which the author also agrees with. This is talked about further in both the discussion and conclusion chapter of this thesis.

Chapter 8

Discussion

In this chapter the author will discuss how well the delivery fulfils the requirements of the original task description, how the testing methodology worked for testing the application, any problems encountered with the project, the results of both the focus group testing and internal testing, and lastly what could've made the project better.

8.1 Deliveries

The main goal of the project was to deliver an application that would help sheep farmers automate the operation of registering their inspection tours to observe the grazing sheep. Based on the requirements of the application that was extracted from the task description and discussions with the contractor (see chapter 3), the application appears to behave as required with a user interface that seems to be satisfactory. The contractor also commented that the feedback for the amount of sheep registered, which makes use of text-to-speech (chap 5.7), was the best implementation of feedback he has experienced so far in all the years of having this task available for students to develop.

There are several decisions specifically to the development and the testing of the application that the author would've liked to have done differently, but more in this in the final section of this chapter.

8.2 Methods

For the development of the application, the author and the contractor met up on a weekly basis to discuss what progress the author had made on the development of the application. During this the contractor could provide feedback on how he wished the application should operate and a discussion could be had. This proved to be extremely valuable for the process of development, and really helped with forming the application. In the last months of the process however, this had to be done over phone/email as of the same situation that limited the focus group scope, and this proved to be a bit difficult. It was hard to really get good feedback as meeting in person is a lot more effective than having to call. Attempting to make notes on the computer while talking on the phone also proved to be difficult.

The focus group methodology was selected to uncover what users of the application actually felt about it, which is important information to have for further development of the product. The focus group was very small and not diverse, which caused the value of the results to be minimal. As it was also mentioned in the testing chapter, the users know the author on a personal level and thus bias is most certainly a factor one has to consider. The author feels however that the methodology for performing focus group testing was done properly, and so a claim could be made that while the focus group itself could've used some work, the methods used are solid.

The usability testing allowed the author to view how a subject interacted with the interface. A very important note was the subjects ability to register sheep without observing the screen of the mobile device. The amount of help they required while using the application was also an important factor to make note of. Both subjects appeared to operate best with the latest implementation of sheep registration, namely the swipe registration. This certainly matches the expectations of the contractor and the author as this is what they concluded with being the most efficient during their internal testing.

The user experience method allowed the subjects to provide anonymous feedback about the application and how they felt like while using it. In general both users agreed on most points, but there was a couple of interesting points where they deviated from one another.

The interview that was held after the usability and user experience testing was quite informative to the author. This allowed for a discussion to take place in which the subjects could talk about possible changes they would have done to the application and allowing the author to explain why certain features behaved the way they are behaving.

The combination of these proved to offer a good insight into how users of the product actually feels about the product. The focus group testing methodology has long been used to test software and it also turned out to be a good choice for this thesis' purpose.

All of the methods worked as intended for the project, and so the author overall is quite pleased.

8.3 Problems Encountered

In this section the author will briefly discuss issues that was encountered while working on this project.

8.3.1 Map accuracy

Map accuracy was a big issue for the application in the beginning. As both Google Maps and Mapbox have rather old and not very detailed maps for the Norwegian outdoors. A lot of time was spent trying to resolve this issue in the beginning phases, without really ever coming close to a better solution. Later in the development stages a solution was finally found, namely using image files provided by Geonorge as overlays to the original maps. This also sadly brought with it several issues.

The images that were provided was all split up into small pieces, which overlapped in strange manners once added to the map of the application, plus it would exceed the limit of how many custom image tiles that was allowed to be used on the map. This was remedied by stitching the images together using GDAL, but this caused the map to be way to large to be uploaded to the mapbox service. The image was then split into smaller parts that was within the limits of mapbox. These map sections however turned out to have black borders around them which would cause the images to cover up other parts of the map. This was remedied by using GDAL once more which finally allowed for a full cover of Norway, but the map was still behaving strangely in the application with it phasing in and out as the user navigated across the borders of the map sections.

In the end only the map section that contained Trondheim was used as this is a prototype and time was limited.

8.3.2 Offline image storage

Another issue that was encountered was the storage of images offline on the device. In theory each observation should have an image of the observation and the user's position, but this proved to be difficult to achieve and so only an image that contained both locations would be saved without any indication of where these locations were.

The images cannot currently be stored locally on the device and are intended to be directly uploaded to the cloud. This of course creates issues for the offline capabilities of the application. Google recommends implementing a sort of queue method that would temporally store the images locally, before being uploaded once a data connection is available.

Given the scope of the application and the time restrictions this was left at this stage.

8.3.3 Android Navigation

Developing for Android can also be generally quite tricky as it has many quirks that one does not find in traditional programming. The major issue that the author encountered was the implementation of their navigation. Android applications are set up by having activities and fragments, in which activities are the major sections of an application, while fragments are minor pieces of the application. Navigating between these can be quite difficult, especially with the addition of the new Android Navigation Component [3].

Preferably an application would have one class of the Navigation Component that contained information for all navigation within the application, however when one uses multiple activities, each activity has to have their own Navigation Component. A lot of time was spent on attempting to fix this by attempting to implement sort of ”sub-components” in the main Navigation Component, but this turned out to not properly function in the end and so each activity has their own Navigation Component file as of this moment.

8.4 Results

The results from the testing matched quite well with expectations that had been made by testing the application with the contractor. It was expected that the swipe method would be the best solution thus far as it was the latest to be developed and best captured the vision that the contractor had. Second place was expected to be the double registration which was developed as a response to issues encountered with the first method, namely single registration. The issue with the single registration was that all the types of sheep were really close to each other and so using it without looking down at the device would be difficult. Also removing sheep on both the double and single registration was difficult as these were buttons which require a great deal of accuracy to press.

8.5 What could've been done differently

This section will go into what the author felt could've/should've been done differently about the whole project. For future developers this can prove to be a good warning about how they should plan out and execute their projects.

8.5.1 Author performance

The author feels like a lot more could've been achieved given the time that was available. This is the first time the author has developed a project over such a long time, especially alone, so knowing how much time to give each section of the process was difficult. In the beginning it was easy to think that a full year is a very long time and so work in the beginning was slow. Undoubtedly the results achieved could've been even better.

8.5.2 External information

A good idea for how to improve the application would be to have some sort of communication with a sheep farmer as they would have invaluable insight into what would be useful. This would also potentially allow for tests to be run alongside the sheep farmer, thus one can have direct input from the customer base.

8.5.3 Report

There should have a been a lot more work done with the report earlier in the process, granted the author had made a basic layout of the report with what should go where, and general notes that would form the different chapters. The transformation of the general notes to a proper report structure however, could've been started a lot earlier and thus saved a lot of time in the later stages.

8.5.4 Known Issues

- Throughout the application both in the actual programming and in the user interface the word report has been misspelled as rapport. This would not be the hardest thing to remedy, at least not on the user interface as it is a matter of changing some strings, but as the application has been labelled as finished all focus was on the report.

- When a user creates a new observation on the map, the observation is not immediately available to click and enter. In order to get it to work one has to reload the map by visiting a different page and then going back to the map, or one can enter the observation from the observation list.

- On the topic of observations, it was also originally intended that the map view for registration of new observations should have some sort of indicator on the map of where the observation would be registered in advance (a sort of ”cross-hair”). This would help users understand better where the observation would end up being registered, but somehow during the implementation process this got neglected. One could also have added a method of changing the size of the observation, thus adding more detail.

- Currently when a user wants to save a map for offline storage, there's no indicator of how large the map will be. It will attempt to store all that the viewer is currently viewing, however mapbox has an internal limit for the sizes and so the download might fail. In a finished solution one would make a better user interface for this whole process and also limit the size that a user can download.

- Throughout the application there is code that can be severely optimised. For an example the mapbox maps can probably be re-used in some manner as to allow the application to not have to reload the map every time it changes views. This would help a lot with the overall performance of the application.

- There's also currently several locations with duplicate code, specifically within the different methods of registration. This is leftover as they are thought to be temporary. In a finished product one of these methods would be chosen to be the best method and thus one would remove the others, which would also remove the duplicate code.

8.5.5 Testing

Testing could've been done throughout the development process of the application. In a way this was done by showing the contractor how the application worked during the weekly meetings, but actually doing full tests both with the contractor and possibly with some external subjects would possibly make the finished product better.

8.5.6 The original plan

The original plan for the project was to have a webpage that went along with the application, thus the focus on having cloud storage of data. The thought was that the webpage would be for when the sheep farmer comes home after an inspection tour and wants to view the data that they have recorded. The farmer would be able to make alterations for the observations, thus one could make smaller notes while on the trip and then go into further detail once at home.

Another feature that would be built into this webpage was the automatic generation of the final season report, which is a combination of all inspection tours performed during a season. The tool that would be used for this would've been Jinja [19]. In order for this to work one would have to create a template that Jinja could use to fill in data provided by the different inspection tour reports. This would automate even more of the farmers work as they have to deliver such a report on a seasonal basis.

8.5.7 Text-to-speech

In the swipe registration method users can double click the screen in order for the device to tell them what they are currently editing. This should potentially have been changed to rather tell them exactly how many sheep have been registered of the current type (e.g. "36 white adults" instead of "editing white adults"). This is a rather minor change but it was something that was picked up on during testing.

One could also look into methods of further implementing text-to-speech, perhaps also have some sort of speech recognition of what the user is saying so that they would not have to manually interact with the interface. However this would possibly be difficult to implement locally as most speech recognition is done in cloud based services.

Chapter 9

Conclusion

The conclusion of this report will be presented in three parts. The first is regarding the application itself and how the author feels about the finished product, the second part will be about the performed tests for the application, and lastly there will be a section for recommendations for future development.

9.1 Application

The application as a whole operates satisfactory to the requirements that were set at the beginning of the fall semester, and so in general the author is quite pleased. Especially considering that the application needs to function properly both when the user has a data connection, as well as when they don't have one. The only aspect of the application that does not completely work offline is the automatic snapshots of the map as a new observation is made. This however was not a part of the original plan and thus not very important to the overall product.

There are certain things that didn't get as far in development as hoped, such as the general map interface, but other parts like the sheep registration method came a long way. It is in the authors belief that the swipe registration is the method that should clearly be developed further.

It is quite clear that the application has a long way to go before it could be released to the public, this is especially true when it comes to the overall user interface which is quite basic and not very intuitive. There's also several interface elements that are sized in abnormal manners compared to other mainstream applications on the market, which also would need to be corrected.

9.2 Performed Tests

Based on the focus group testing with the following interview combined with the internal testing, it became quite clear that the swipe method has the most potential amongst the current registration methods. Both subjects felt that it was easy to use without watching the screen of the device, and this was confirmed by observations done by the author during the usability testing. As it was quite simplistic with each direction corresponding to a given action, users mastered it quite quickly and became efficient with it. In general both subjects also agreed on that the application needs a lot of user interface improvements as this negatively affects the overall user experience.

Longer and more complicated tests with actual sheep farmers would be ideal for discovering where to go from this point.

9.3 Recommendations for future development

For future developers there are several recommendations that the author would like to make, and these are as follows:

- **Google Maps/Mapbox/OpenStreetMap** The author never got around to testing map tiling with Google Maps, but if it does not work as well as it turned out to do with Mapbox, then certainly Mapbox is a good alternative. The problem here is with updating the map to reflect the newest data from Geonorge as one would have to upload the new map to Mapbox every time, however they do offer an API for this so in theory a server could potentially request data from Geonorge, process it and upload it. Another alternative is attempting to utilise OpenStreetMap [10] directly, which several styles that Mapbox offers are built on, to create a map service specific for this application. This would require a lot more work, and would certainly greatly expand the scope of the overall task.

- **Feedback: Text-To-Speech** Text-To-Speech is definitely something that future developers should implement into their solutions, as this proved to be a very effective method of letting the user know what actions they are performing on the device while registering animals without having to put down their binoculars.

- **Mobile solutions** The author would definitely recommend developing an application for both Android and iOS in order to cover most of the market. The reason this is recommended over making a webpage solution that would work on both devices is that these operating systems have a lot of built-in functionality that makes such an application easier to design and implement. Especially when developing a product that needs to operate as intended both online and offline. A webpage could be made as an addition to the general package as a way to view/extract the data that has been registered with the mobile device (and potentially automate the generation of the seasonal report, see chap 8.5.6).

- **Google Firebase** Google Firebase turned out to be a very good choice for the handling of storage of data and taking care of authentication functionality. It works excellently cross platform and can handle the whole offline storage of information without a problem. Future developers should look into how to make it only synchronise data linked to one account, as for the moment it appears as all data for all users is synchronised, which will take up a lot of space on the device eventually. Looking into creating a local queue functionality for the uploading of images to the Cloud Storage would also be a good idea, if the plan is to add images to the project (images of the map or possibly images taken in an observation).

- **Swipe registration** As mentioned previously the author believes strongly that the swipe registration method is the method that should be developed further. Possibly one could experiment here with adding shortcuts as subject 1 of the focus group suggested for quickly switching between colours of sheep, or one could possibly add a toggle-switch between adults/lambs somewhere in the interface so that one could easily register both adults and lambs of different colours.

- **Lamb calculator** In the requirements chapter for recording sheep observations (chap 3.1.2), it was mentioned that ewes with lambs have a coloured tie that corresponds to how many lambs they should have. So for future versions it could be a good idea to have a method of registering this and having the application automatically calculate how many lambs should be in the herd, and then offer a comparison to what the farmer actually registers while observing the flock.

Bibliography

[1] Allma. Om allma. https://www.allma.no/om-allma. Accessed: 2020-03-19.

[2] Allskog. Allma. https://www.allskog.no/allma. Accessed: 2020-03-19.

[3] Android. Get started with the navigation component. https://developer.android.com/guide/navigation/navigation-getting-started. Accessed: 2020-05-07.

[4] Android/Google. About the android open source project. https://source.android.com/. Accessed: 2020-04-13.

[5] Apple. Start developing ios apps (swift). https://developer.apple.com/library/archive/referencelibrary/GettingStarted/DevelopiOSAppsSwift/. Accessed: 2020-04-23.

[6] Jakob Nielsen Don Norman. The definition of user experience (ux). https://www.nngroup.com/articles/definition-user-experience/. Accessed: 2020-05-01.

[7] DrawIO. Drawio. https://app.diagrams.net/. Accessed: 2020-05-15.

[8] Findmy. Findmy om oss. https://www.findmy.no/nb/om-oss. Accessed: 2020-03-04.

[9] Findmy. Findmy product. https://www.findmy.no/nb/produkt. Accessed: 2020-03-04.

[10] OpenStreetMap Foundation. Openstreetmap. https://wiki.osmfoundation.org/wiki/Main_Page. Accessed: 2020-05-08.

[11] gdal. Gdal. https://gdal.org/. Accessed: 2020-04-25.

[12] Geonorge. N50 raster (utm33) - rutevis. https://kartkatalog.geonorge.no/metadata/n50-raster-utm33-rutevis/

1a7d4919-fc5d-4278-9ae9-45ee23f26808. Accessed: 2020-04-25.

[13] Geonorge. Om geonorge. https://www.geonorge.no/aktuelt/om-geonorge/. Accessed: 2020-04-23.

[14] Google. android studio. https://developer.android.com/studio. Accessed: 2020-04-25.

[15] Google. Firebase helps mobile and web app teams succeed. https://firebase.google.com/. Accessed: 2020-04-24.

[16] Google. Maps sdk for android. https://developers.google.com/maps/documentation/android-sdk/intro. Accessed: 2020-04-23.

[17] Anders Horntvedt. Marit mjøen solem i findmy begynte med sporing av sauer. nå brukes bjellene i løveområdene i kenya. https://finansavisen.no/nyheter/teknologi/2020/02/11/7496663/marit-mjoen-solem-i-findmy-begynte-med-sporing-av-sauer.-na-brukes-bjellene-i-loveomradene-i-kenya, 02 20. Accessed: 2020-03-04.

[18] IDC. Mobile operating system market share worldwide. https://www.idc.com/promo/smartphone-market-share/os. Accessed: 2020-04-23.

[19] Jinja. Jinja. https://jinja.palletsprojects.com/. Accessed: 2020-05-07.

[20] JSON. Introducing json. https://www.json.org/json-en.html. Accessed: 2020-04-23.

[21] Mapbox. Mapbox pricing. https://www.mapbox.com/pricing/. Accessed: 2020-04-23.

[22] Mapbox. Maps and location for developers. https://www.mapbox.com/. Accessed: 2020-04-23.

[23] Mapbox. Maps sdk for android. https://docs.mapbox.com/android/maps/overview/. Accessed: 2020-04-23.

[24] Vasilis Papakostopoulos Nicolas Marmaras, George Poulakakis. Ergonomic design in ancient greece. https://www.sciencedirect.com/science/article/abs/pii/S0003687098000507. Accessed: 2020-05-01.

[25] Nofence. Give your animals access to better and more varied pastures. https://nofence.no/about-the-product/. Accessed: 2020-03-05.

[26] Gary Perlman. User interface usability evaluation with web-based questionnaires. http://garyperlman.com/quest/. Accessed: 2020-05-06.

[27] Mishaal Rahman. Android version distribution statistics will now only be available in android studio. https://www.xda-developers.com/android-version-distribution-statistics-android-studio/. Accessed: 2020-04-25.

[28] Den Norske Regjering. Kartverket. https://www.regjeringen.no/no/dep/kmd/org/etater-og-virksomheter-under-kommunal--og-moderniseringsdepartementet/underliggende-etater/kartverket/id85701/. Accessed: 2020-04-23.

[29] Shiip. Morgendagens løsning. https://www.shiip.no/produkt. Accessed: 2020-03-05.

[30] Shiip. Shiip your sheep. https://www.shiip.no/. Accessed: 2020-03-05.

[31] Smartbjella. Om smartbjella. https://smartbjella.no/om-smartbjella/. Accessed: 2020-03-04.

[32] SmartBjella. Om sporing av sau og rein. https://smartbjella.no/om-sporing-av-sau-og-rein/. Accessed: 2020-03-19.

[33] SQLite. What is sqlite? https://www.sqlite.org/index.html. Accessed: 2020-04-24.

[34] Telespor. Nye kart. http://telespor.no/nye-kart/. Accessed: 2020-03-05.

[35] Telespor. Produkt radiobjella. https://telespor.no/produkt/. Accessed: 2020-03-05.

[36] Marcin Treder. The history of user experience design. https://medium.com/@marcintreder/the-history-of-user-experience-design-5d87d1f81f5a. Accessed: 2020-05-01.

[37] Ari Wubbold. The birthplace of focus groups. https://dhmresearch.com/custom-research/2012-09-04/the-birthplace-of-focus-groups.html. Accessed: 2020-05-01.

[38] Svein Ø. Sveins blogg - e-bjøller. https://sveino.blogspot.com/2014/06/e-bjller.html. Accessed: 2020-03-04.

NTNU

Norwegian University of
Science and Technology