Bachelor Project Final Report TI3405

AGRIAPP

Authors:

Ka Ho Man 1294415
Andy Chiu 1519360
Julian Faber 4189736

TU Delft Coach:

Tomas Klos

AgriPlace supervisor:

Maarten den Uijl

Bachelor Project coordinators:

Martha Larson
Felienne Hermans

July 2, 2015

Foreword

For the last project in the Computer science bachelor program, every student needs to form their own team and choose a software project where they can apply their acquired skills professionally at a real company. Hence there was a strong preference for choosing a project not related to the TU Delft. AgriPlace offered an interesting project that would simplify the administrative work of growers. This seemed like a very refreshing project with a refreshing background to apply our software engineering skills. We were asked to build a mobile application next to an existing web platform. This final report is written to document the entire process from research up to the realization of the actual software solution. We would like to thank Tomas Klos for being our TU coach and providing us with helpful tips and feedback during weekly meetings. Equally many thanks go to our client's product owner Maarten den Uijl who created this project and made effort that the team was provided everything they needed. Finally, we feel grateful for having Martha Larson and Felienne Hermans as coordinators that solved problems we've faced during the project.

Summary

The People 4 Earth Foundation strive to create food production and trade more sustainable. They do this by creating a common language for sustainability assessments. These assessments are filled in by growers who want to certify themselves according to a certain standard. On AgriPlace, growers can easily manage their assessments and share them with interested parties. This is an improved and simplified way of doing assessments, because the questions and documents that need to be provided for an assessment contain much overlap. AgriPlace can detect overlap and fill in parts of an assessment automatically with data that has previously been provided.

The team of students from TU Delft are asked to create a mobile application that can be used for doing assessments, even when no internet connection is available. In this report, this application will be referred to as the AgriApp. Users must be able to perform assessments offline and synchronize data with the server when they have an internet connection. Research has been done for deciding which development framework to use, how the user should interact with the system and an initial user interface design is given. Maintainability is one of the most important factors, so that is the reason why the application is created using PhoneGap, a cross-platform development framework. PhoneGap uses standard web technologies, which is something the AgriPlace developers have a lot of experience in. Furthermore, local data is stored in an SQLite database and the containing data will be synchronized with the server using the team's own implementation. At the time of writing, synchronization does not yet work, because the AgriPlace developers still need to provide the required end points for communication with the server.

Table of Contents

1. Introduction

Food is considered an essential part of life. Today's people are more aware of the things they eat, but what is less well-known is the impact the production and consumption of food has on the world's resources. Analyses question the long-term sustainability of the current trend of food production and consumption. A report (Freibauer et al., 2011) concluded that the world's resources are being used at an alarming rate, much higher than they can be naturally replenished. In the future, food is expected to be even more inaccessible to the poor. Fortunately there are many growers producing their crops in a sustainable manner. Companies that buy their crops will sometimes demand that they comply with several standards before the product is placed on the market. These standards prescribe a range of policies and practices that should be in place in order to receive certification. The process for acquiring a certificate as proof of compliance is very time consuming and information is scattered. Growers usually do not like the associated administrative burden and do not consider themselves as 'desk' workers.

The People 4 Earth foundation[1] (P4EF) is a Dutch organization with a passion to make food production and trade more sustainable. They founded AgriPlace[2], a start-up with the purpose of lowering the barriers to transparent and sustainable food production. Growers are provided with the necessary guidelines for each standard and to simplify the certification of their products. AgriPlace is accessible through the website and a team of students is invited to develop a mobile version for offline and online usage. This final report is dedicated to the development of the mobile application and the process during the project. This gives the AgriPlace team insight into the technical specifications, which they can use for continuing the development of the application.

The first two weeks were used for research on the problem, the background and a possible solution, these results can be found in Chapter 2. Chapter 3 contains requirements that the proposed solution must meet. Several development frameworks were then discussed with the client. A use case for interacting with the application was created, together with a graphical user interface (GUI) mockup to give an impression of the final product, which is the main purpose of Chapter 4. Chapter 5 will provide an overview of the final product and contain all details regarding the implementation. Much attention will be given to changes that have occurred and the actions that were taken as a response. Testing of the application is described in Chapter 6. When the implementation phase was over, the final product and the process were evaluated in Chapter 7. Finally, we end this report by giving some recommendations and a conclusion in Chapter 8 and 9.

[1] http://www.people4earth.org/
[2] http://www.agriplace.com

2. Problem Analysis

This chapter is dedicated to the research phase of the project where a thorough understanding of the problem is created. Once the problem has been defined in detail, an analysis is performed to see what could be a fitting solution to solve the problem. The team is given twelve weeks to solve the problem, which is a relatively short time. This why an appropriate software development methodology is chosen in order to maximize the team's productivity and to give insight in how the team cooperated.

2.1.Problem background

Growers who wish to place their food products on the market are required to comply with a number of standards. Standards prescribe a range of policies and practices that should be in place before growers receive a certificate. A grower can start an assessment which consists of answering a number of questions and providing evidence documents. An auditor reviews the assessment and issues the certificate to the grower so that they can show their compliance with the standard. Doing multiple assessments can be very time consuming. Fortunately, the People 4 Earth foundation realized that a significant overlap with different assessments existed, which makes it possible to simplify the process. They created AgriPlace, an online platform for collecting and sharing farm data. Growers working with a number of standards for their product will benefit much from the simplified certification process when using AgriPlace. User entered information about the organization and products is saved which can be easily reused for filling in future assessments, this saves time and allows the growers to spend more time in doing what they do best.

2.2. Problem definition

AgriPlace is accessible through a web browser connected to the internet and is expected to be deployed in different countries in the coming years. Eventually, AgriPlace will expand to include third world countries too. An issue concerning growers in developing countries as well as in some remote places is the low coverage and internet connectivity. Growers need to be able to fill in assessments outside, even when no internet connection is available, since that is where they spend most of their daily time. Sometimes certain evidence documents are not present in digital form, so growers need to scan or take a picture of these documents first before they can upload them. This extra step is considered cumbersome by the less technically skilled growers.

2.3. Project analysis

Although internet connectivity is not available everywhere, smartphones that are able to run third party applications are becoming more widespread. Hence, a software solution for smartphones is recommended to perform offline assessments outside. The filled in assessment including evidence documents must be stored locally until an internet connection becomes available and synchronization with the server can take place. The application should be able to use the phone's resources, such as accessing the file system and the camera module for taking pictures of crops and documents. The usage, look and feel of the application should resemble the browser version as much as possible. A project description formulated by the client can be found in Appendix A.

2.4. Software development methodology

An agile software development methodology named SCRUM[3] is used to anticipate any changes in requirements and to maximize team efficiency. SCRUM is used a lot in today's software industry, including our own client. Adopting the same software methodology as the client will lead to easier project synchronization which will be discussed below.

Three sprints will be spent on actual development and every sprint has a duration of two weeks. A sprint is a time-box where tasks will be planned into, so every team member needs to complete their task in every time-box. At the end of each sprint, a particular feature of the application should be ready for demonstration. In our team, Ka Ho will be the one serving as the product owner; a product owner represents the interests of the stakeholders. He defines the product backlog with all the desired features by the client and the estimated time for development, this can be found in Appendix B. Andy will be assigned the role of Scrum Master; a Scrum Master is the leader of the Scrum process. He'll make sure the SCRUM process is followed and that everybody is provided everything they need so that they are able to complete their tasks on time. Finally, Julian will be serving as the Lead Programmer. As a Lead Programmer he's responsible for the architecture of the software program and is able to estimate the required time best.

The client also has a sprint duration of two weeks with a demonstration on the second Wednesday. AgriPlace is continuously improving and the team is invited to join these demonstrations so that they are up to date with the latest development changes. Changes in requirements or use cases can then be detected and incorporated in the mobile application. This is important, because the web and mobile application should have a consistent interface and usage pattern. The benefit of having the same sprint duration is so that a demonstration of the mobile application can be given as well on the same Wednesday, this is called a synchronized heartbeat. This is the main reason why the team has decided to meet up in Amsterdam at least once a week on Wednesday, to ask questions and give a demonstration.

[3] http://www.scrum.nl/site/Scrum-Begrippen-agile-scrum

For the rest of the week the team will work independently on available work spaces at the TU Delft. The team understands the importance of meeting up regularly with the client, but they feel they can be more productive if the long travel to Amsterdam can be omitted.

3. Requirements

This section discusses the list of requirements that has been gathered together with the client, according to the MoSCoW-method[4]. This list gives a clear picture which functionalities the client values most and which should only be implemented when extra time is available. Must Haves are features that are minimally required before the application will be accepted by the client. Should Haves are highly appreciable, but not required for the project to succeed. The lowest priority features are the Could Haves, these add little value and thus are only to be implemented when extra time is available. Won't haves are not implemented at all. The non-functional requirements describe how a system is supposed to be in contrast to the functional requirements that describe what the system should do. Some assumptions in the form of constraints are required for certain features to work.

3.1.Functional requirements

Must have

M1. Users can log in using their AgriPlace account
M2. Users can log out
M3. Users can start an offline assessment and gather the evidence documents needed for this assessment
M4. An open assessment can be resumed offline.
M5. Documents required for an assessment can be added.
M6. Photos required for an assessment can be added or taken directly using the phone's camera application.
M7. An open assessment can be stored locally on the smartphone.
M8. A locally stored assessment can be synchronized manually when internet connection is available.
M9. The list of crops can be edited.
M10. The list of crops can be viewed
M11. It is visible which offline assessments have been synchronized

Should have

M1. Questions can be answered with user input
M2. Questions can be auto filled using evidence documents

Could have

C1. The user can see all issued certificates.

[4] https://en.wikipedia.org/wiki/MoSCoW_method

C2. Users can see their own profile.
C3. The profile can be edited.
C4. Access to archive
C5. Issued certificates for a specific product are visible.

Won't have

W1. Crops cannot be removed.
W2. Assessments cannot be removed.
W3. Evidence Documents cannot be removed.

3.2.Non-functional requirements

NF1. The application must have an intuitive interface which is easy to use and understand without any training.
NF2. The code must be of sufficient quality and provide documentation so that it can be reused by other programmers.
NF3. Navigating through the interface must feel responsive and smooth.
NF4. The user interface should give the same look and feel as the website.
NF5. Users should follow the same steps taken to perform a task on the website as much as possible.
NF6. The application must be able to run on Android OS and possible iOS.
NF7. Communication between the server and the smartphone is encrypted.

3.3.Constraints

CON1. The user must have an AgriPlace Account.
CON2. Internet connection is required to synchronize.

4. Design

This chapter explains the main design choices divided in a technical part (section 4.1) and a functional part (section 4.2 and section 4.3). The technical part contains the research done on the different development framework that can be used to realize an application that should be available on multiple platforms. How the users interacts with the system will be shown in a use case diagram and an initial design of the graphical user interface is given so that it can be used as reference during the development.

4.1.Mobile phone application development

Developing applications for a mobile phone requires a development framework that will translate the code to an application. Native solutions are provided by the vendors of a certain mobile platform, they provide all the tools necessary to create a rich application for their own platform only. This would seem like a logical choice if the application is only intended to run on one particular platform, but that is not the case. Therefore hybrid solutions are also considered, which are easier to port to multiple platforms. In this section, we compare the multiple hybrid and native solutions.

4.1.1. Solutions

There are two kinds of solutions for implementing the software: Native and Hybrid[5]. A native solution means every platform needs to have its own version of the software. For example, if you would like to make an application for Android phones and iPhones, you need to have two applications to support two kinds of phones. A hybrid solution allows you to reuse a major part of the code and build an application for different platforms. Next, the advantages (+) and disadvantages (-) of a native and hybrid solution are shown.

Native:

+ The application has better performance.
+ The application will be more responsive.
+ It is easier to access device functionalities like Camera and GPS.
- Every platform needs its own version of the same application.
- The number of users you can reach is limited to the users of a specific platform, because not everybody is using the same platform.

[5] http://www.mobiloud.com/blog/2012/06/native-web-or-hybrid-apps/

- More difficult to maintain the application when multiple versions exist for the different platforms

Hybrid:

+ Reuse a major part of the code and build the application for different platforms
+ One programming environment for building on multiple platforms
+ A multi-platform application will be accessible to a wider audience
+ Investment of application cheaper in comparison to other solution(s)
- The application has lower performance compared to native
- The application will be less responsive compared to native
- If you would like to develop in iOS you will need to have an Apple device with OS X and an expensive yearly developer license (same goes for native iOS)
- The accessibility of certain features on the phone is limited to the available plug-ins.

This shows that both solutions have their own advantages. If the focus is placed on application performance, native will be a better solution than hybrid which can be read in a comparison article (Sommer & Krusche, 2013). However, a note must be made that hybrid solutions are narrowing this gap (Charland & Nitobi, 2011). If you are placing the focus on greater reach of users, it is better to use a hybrid solution, and it would be cheaper to maintain. So the three big factors for deciding which solution is best are: Performance, reach and maintainability. When going for a native framework, the advantages and disadvantages between the tools available are not much different than listed in section 4.1.1. so these are not investigated any further.

4.1.2. Hybrid Frameworks

In our search[6] for hybrid frameworks, the AgriPlace development team recommended the PhoneGap framework, but multiple frameworks have been considered and each of their unique properties are listed. Three different frameworks are being used by a large community, these are: PhoneGap[7], Xamarin[8] and Appcelerator Titanium[9]. The advantages and disadvantages of each hybrid framework is shown below. In the next section the decision will be made based on the advantages and disadvantages of the solutions (section 4.1.1.) and the hybrid frameworks.

[6] http://www.optimusinfo.com/cross-platform-framework-comparison-xamarin-vs-titanium-vs-phonegap/
[7] http://www.phonegap.com
[8] http://www.xamarin.com
[9] http://www.appcelerator.com/

PhoneGap:

+ It is free to use.
+ Can reuse almost 100 percent of the code for different platforms
+ Programming using standard web technologies[10] (HTML, CSS, and JavaScript).
- Cannot reach native performance.
- Hard to create large applications with JavaScript, compared to strongly typed languages like C#

Xamarin:

+ Compiled code is platform specific, can achieve high performance
+ Uses C# which is easier to code
+ Has visual design environment for android and iOS
- Requires a monthly fee for further development.
- Less reusable code compared to PhoneGap and Appcelerator

Appcelerator:

+ Compiled code is a combination of native and JavaSript, performance improved over PhoneGap
+ The resulting UI look-and-feel is close to native
- Requires most expensive licensing fees
- Hard to create large applications with JavaScript
- Developers must learn the Titanium API

4.1.3. Framework choice

The development team of this project suggested using a native solution with software tool Eclipse. The reasons are:

- The client would like to have application(s) that could support both Android an iOS, but the one that needs to be finished first is the Android version. So with the time the development team has, it is not enough to make both versions even if they use a hybrid solution.

- The development team is more experienced with programming in an object orientated language than web programming. So if they need to use web programming or other languages, they will need to spend more time to learn the language.

[10] http://www.w3schools.com/

- The client also would like to have a mobile application that feels responsive and smooth. So a natively programmed application has a better responsiveness and runs faster than an application that is programmed using a hybrid solution.

- One of the disadvantages of hybrid solution is bad performance. And bad performance is in conflict with one of the non-functional requirements. So the development team suggested not to use a hybrid solution.

The client did not agree with the development team's proposal to use a native solution. Their reasons are:

- The AgriPlace developers are experienced with web programming. If the TU team made an application that is programmed in java and the client would like to use the application for further development. Then it is logical to use web programming for the mobile application.

- If the team is going to use a web programming language to implement the application. The AgriPlace developers can fully support us if we have problems. If the team is going to use the Java language and they get stuck on something, the AgriPlace developers cannot provide support.

- It would take much effort to maintain the application for different platforms if a native solution is used.

After the suggestion and the discussion with the client, the development team has agreed to use a hybrid solution and the software tool PhoneGap. The most important factor for making this decision is maintainability. If the eventual prototype is to be developed further by AgriPlace, then it makes sense to choose a programming environment they are experienced in. Of the three hybrid solution (PhoneGap, Xamarin and Titanium), only PhoneGap and Titanium use web technologies. PhoneGap was preferred over Titanium, because of the higher code reusability. Also, the AgriPlace development team has made an earlier attempt to create a PhoneGap application. However, this choice will lead to decreased performance.

4.2. Use case diagram

A use case diagram[11] has been made to show the different actions the user can perform on the application. The number of actions the user can perform is limited. So instead of creating use case diagrams of single actions, a unified use case diagram for all actions is made (figure 1).

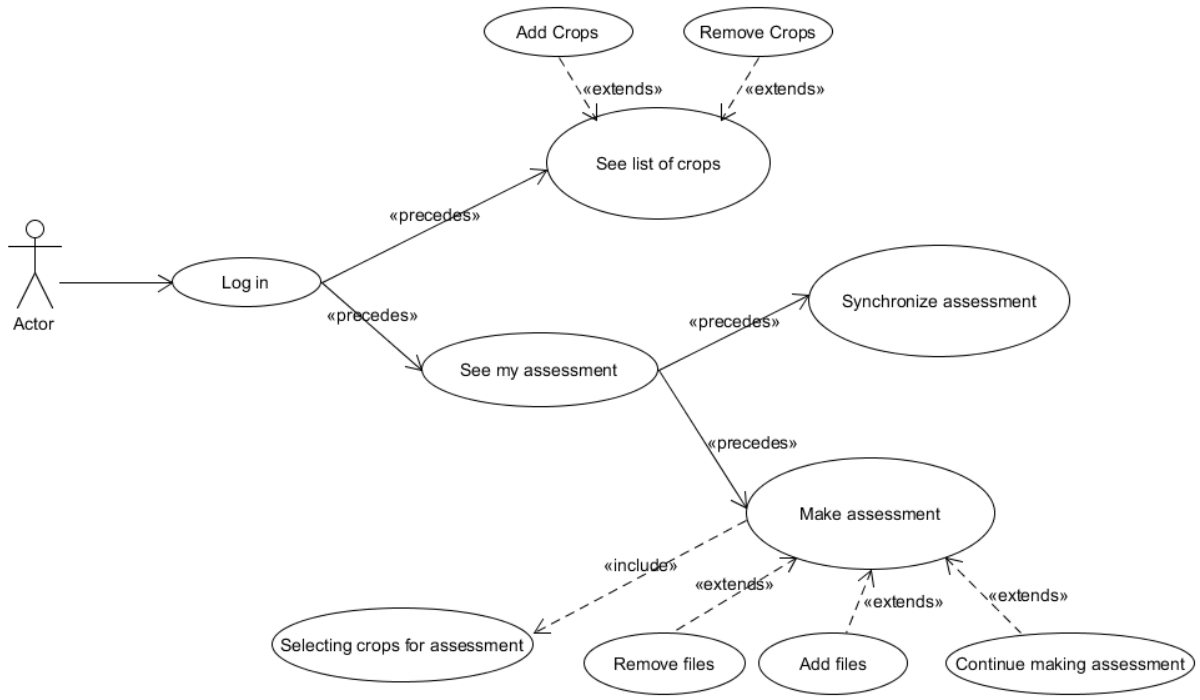

Figure 1. Unified use case diagram of mobile application.

The diagram shows all the actions that can be performed in the mobile application. First you need to log in to your account. After logging in, you can choose to update (adding crops or removing crops) your list of crops or make an assessment. When you would like to make an assessment, you need to choose the crops you want to perform the assessment on. An assessment consists of providing evidence documents in the form of files and answering questions. Files can be added and removed by the user. Assessment can be continued at a later time if not all questions have been answer or not all evidence documents have been provided. After every question is answered, it is possible to synchronize with the server when an internet connection is available.

[11] http://agilemodeling.com/artifacts/useCaseDiagram.htm

4.3. Mockup

In this part, the mockup for the front end is shown. The user interface is designed with keeping in mind that growers are generally less technically skilled. Screen elements are made simple and large to accommodate this. People have limitations on their short-term memory and they have to deal with more distractions in a mobile environment than behind a desktop (Gong & Tarasewich, 2004). That is why each screen shown to the user must be kept simple and easy to understand. This mockup is a prototype to show what the product is going to look like and how it will function.

Figure 2. Log in screen and home screen

When the application starts, you will see the log in screen (figure 2). To be able to go to your account, you need to fill in your username and password. If the username or password is not correct, a red text with “username or password is not correct” will appear. If it is correct, the home screen is shown with three options (figure 2): Crops, assessment and options. There is no screenshot showed of the option screen, because the screen is blank. So we will not give further description of the option screen in this section.

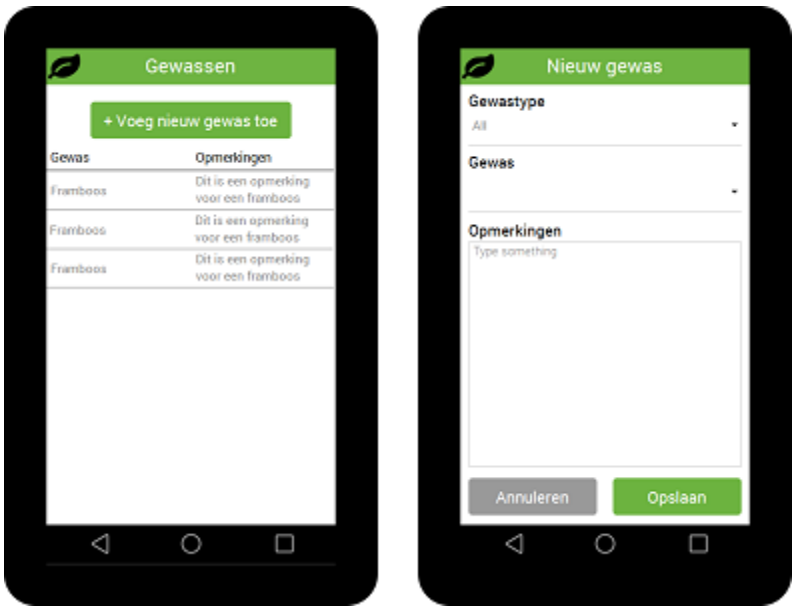

Figure 3. Crops screen and new crop screen

When pressing the Crops button, you will see a list of crops (figure 3) and the comments regarding a certain crop. If you would like to add a crop, you can press the button “add crop” and go to the new crop screen (figure 3) to choose the crop you would like to add. To limit the list of all crops, you can filter them using the dropdown menu with crop types. When you have chosen your crop, you are also able to add any comments for that crop. Finally, press the save button so that the crop will be added to the list on the crops screen. The cancel button on the new crop screen is used to return to your list of crops when you change your mind about adding a crop.

Figure 4. Assessment part 1: main assessment, the standards, choose crops and evidence screen

The other option of the home screen is the assessment button, you will go to the main screen of your assessments (figure 4). There are two lists: the open – and closed assessments. Open assessments are assessments which are not ready for certification or not completely filled in. Closed assessments are the ones which are already being reviewed and should not be edited. There are also two buttons: “start new assessment” and “synchronize”. The synchronize button is used to upload any evidence documents to server that have been added using the phone. When you are starting a new assessment, first you need to choose a standard. After you choose the standard, you can choose your organization’s crops that need certification. Then you will see the evidence screen where documents like files, pictures and text can be added. All requested evidence documents need to be provided in order to get certified.

Figure 5. Assessment part 2: questionnaire, upload files, added files and finished screen

After the evidence documents comes the questionnaire, one question will be shown on the screen at once. You can navigate through the questions using the next – and previous button. If the question requires an evidence document that has been provided earlier, a reference to that document will automatically be shown. Otherwise, the user can still add it. The names of added documents are shown in a list. When you have answered every question in the assessment, you will see a screen saying “the assessment is finished”. Then you will be led to the main assessment screen, so you can choose to do another assessment or to synchronize the assessment. When you synchronize assessments, the answers and all evidence documents that you have provided to that assessment will be merged with the server of AgriPlace.

5. Implementation

After the product has been thoroughly described, it is about time to start implementing. In the implementation phase, changes have been made depending on the situation. An overview of the resulting final product is given and a comparison is made with the initial design. If there are changes, argumentation will be provided to support the decisions that led to those changes. Some helpful tools and challenging aspects of the application that require more explanation are also discussed.

5.1.Final product

In this section we are about to show the final product. We are going to use the screenshots from the final product to show what the differences are in comparison with the mockup version in our research phase. The detailed argumentation of the changes will be discussed in section 5.2.

5.1.1. Main screens

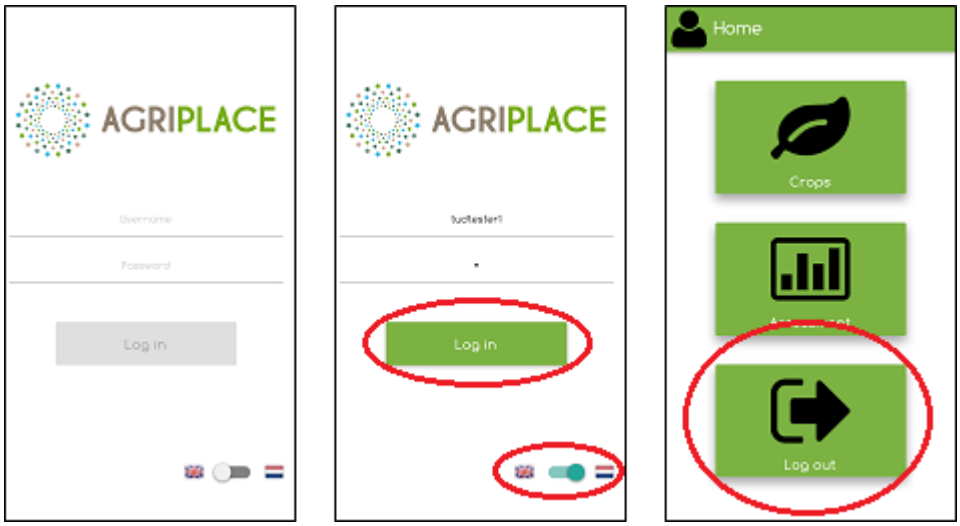

Figure 6. Log in screen a, Log in screen b and home screen

The log in screen is basically the same, but there are some minor changes: 1. when the application is started, the “log in” button is deactivated (figure 6: log in screen a). It will be activated (log in screen b) when a username and password is filled in. You are still not able to log in when the username or password is incorrect. 2. There is a toggle button for changing languages, because the client would like

to have an option to change languages in the future. Momentarily it has no function, so this feature will not have an argumentation in the next section. In the home screen, the option button has been replaced with the “log out” button. Because we have no other options than changing languages, so we placed the switch languages option in the log in screen. The “log out” button has been placed in the big button instead of a small button in the upper right corner.

5.1.2. Crops screens

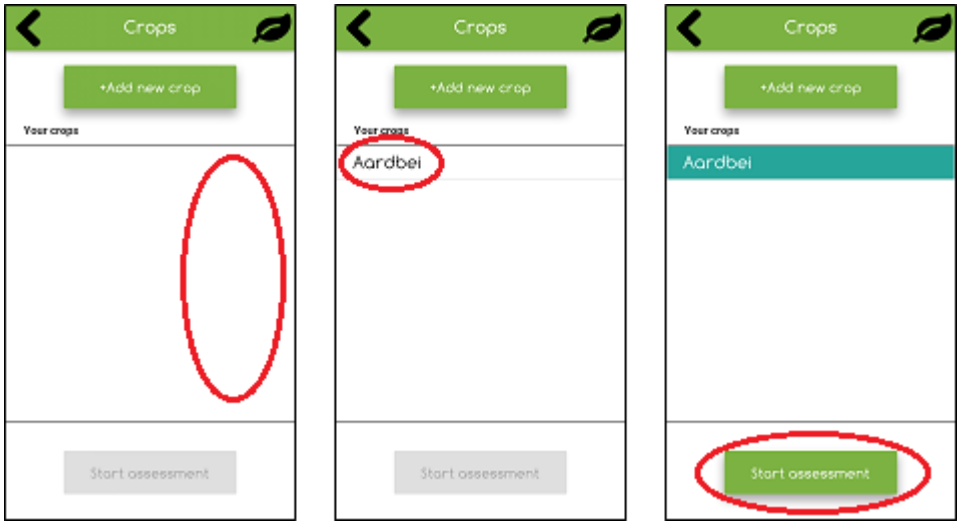

Figure 7. Crops screen: Crops screen when added a crop and Crops screen when selected a crop

Now we are about to show what the differences are at the crops screen. At the left side of the screenshots (figure 7) the users will see a list of crops that have been added to your list, what differs with the mockup in this part is that the comments of the crops have been removed. This is removed because the website version does not support comments on crops anymore, so the development team also decided to exclude this from the application. The second screenshot in figure 7 shows the result of a crop being added to the list. The name of the crop will be shown on the list. It is possible to directly make an assessment when one or multiple crops have been selected, as shown in the third screenshot of figure 7. When a crops is selected, the “start assessment” button will be active and the user can directly proceed to the assessment by clicking on that button.

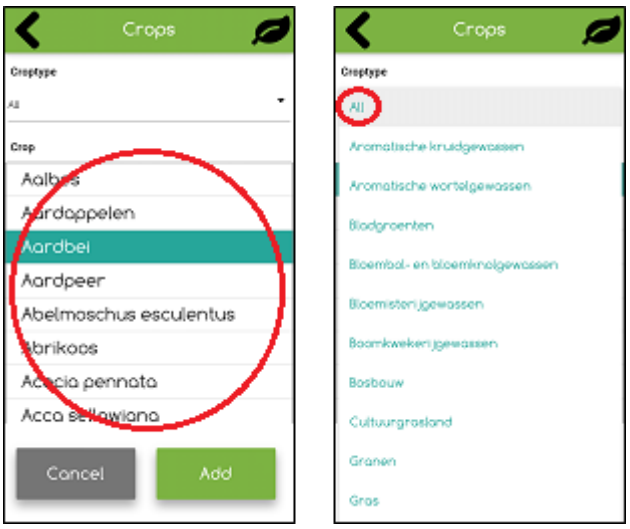

Figure 8. Adding a crop: Add crops screen and choose crops type

In the crops screen the user is also able to add a crop. When the user starts to add a crop, a list of all available crops is shown, as can be seen on the left side of figure 8. The user can also filter the crops by their crop type. This can be done by selecting a crop type from the drop-down list, which will then only show the crops with that crop type.

5.1.3. Assessment screens

In this section we will show the biggest part of our product; the assessments. Because this is a big part of our product we will split it into two parts: the start assessment part and the evidence part. There should also be a questionnaires part, but this was not implemented due to the limited time available.

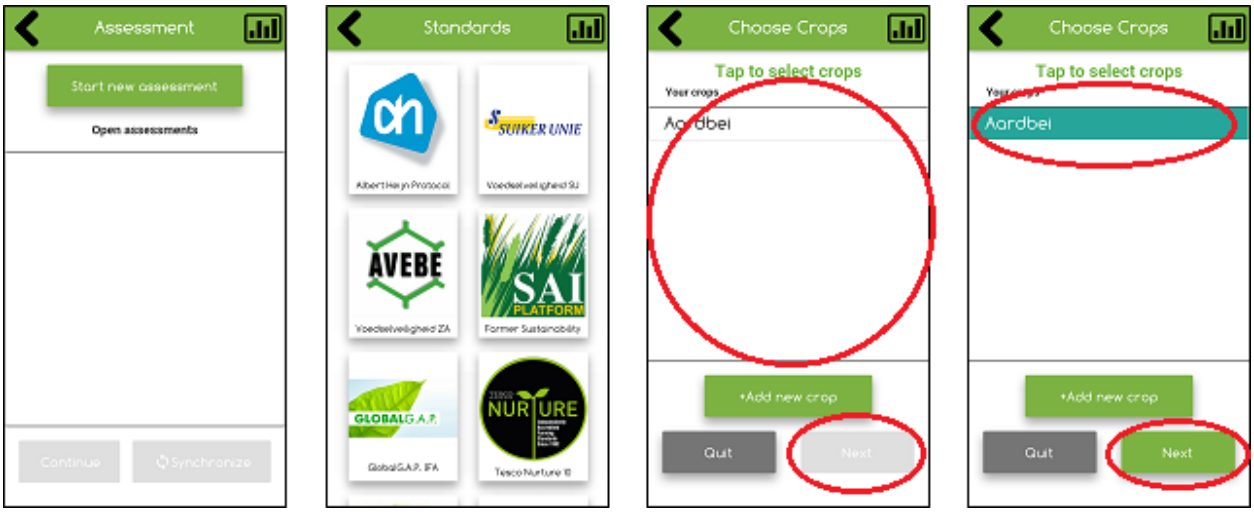

Figure 9. Assessment part 1: Main assessment screen, standard screen, choose crop screen and choose crop screen when selected crops.

The first part of the assessment, the flow of the screens and the options, is basically the same as the mockup version. So we will only discuss what differs from the mockup version. At the main screen the "continue" and "synchronize" buttons are deactivated (shown in grey). In the mockup version they are shown in green, but they did nothing when the user pressed them. Other differences of these two buttons will be discussed later to make the flow of the screens chronological. The standard screen does not differ with the mockup version so we will skip this screen and go to the choose crops screen. In the mockup selecting a crop on the choose crops screen would add it to a "selected crops" box. Instead of that, we have chose to expand the list and let the user be able to select multiple items in that list. Another change is that the "next" button will be activated when a crop has been selected, as shown on the right side of figure 9.

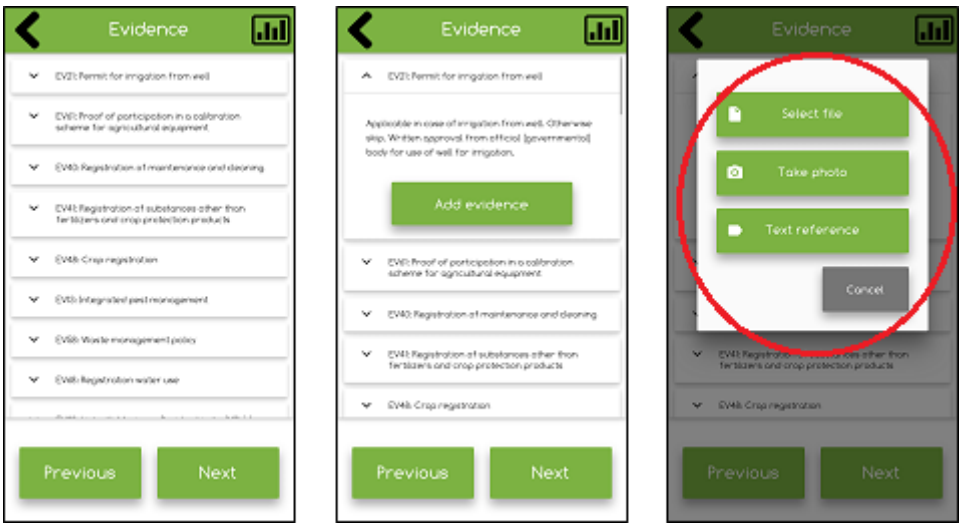

Figure 10. Evidence part 1: Main evidence screen, evidence item expanded a text and selecting an attachment type as evidence.

The second part of the assessement: adding evidence documents. The screenshots shown in figure 10 do not really differ from the mockup version that the team has made. When you see the evidence screen, you will see a list of questions to add evidence as shown on the left side of figure 10. The user can expand to see the whole question, and is able to choose to add evidence or not. If the user chooses te add evidence, the screen will show a pop-up screen with the options “select file”, “take Photo”, “text reference” or cancel to not to add evidence.

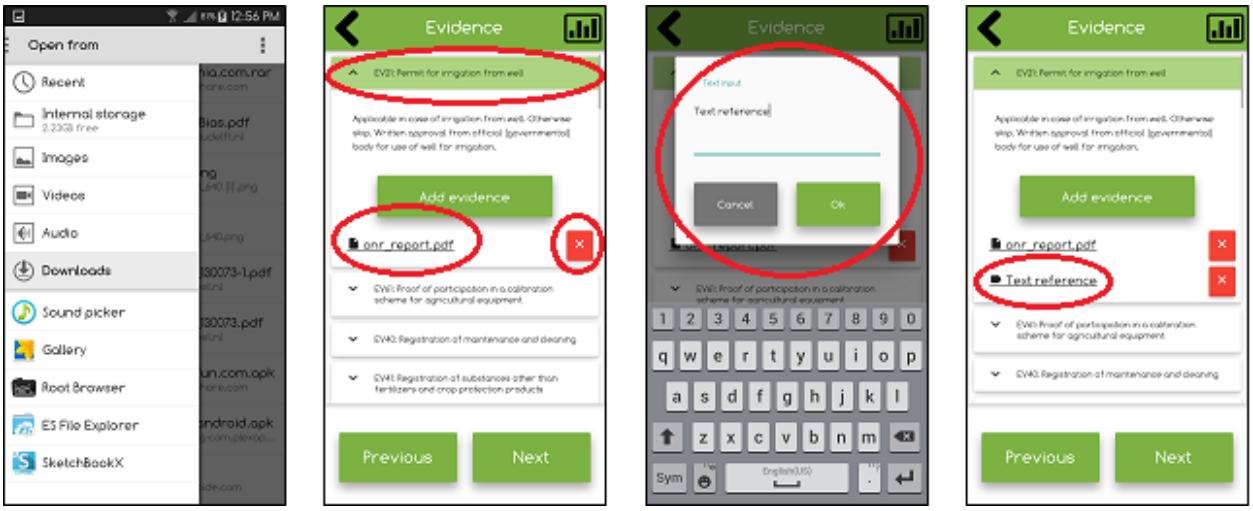

Figure 11. Evidence part 2: Select a file, added evidence file, add a text reference and added text reference.

If the user chooses to add a file, the application will let the user browse the file system of the mobile phone. When the needed file is selected, the file name will be shown under the button add evidence. If the user would like to remove the file, they simply press the red button to remove the file. Also when a file is added, the background of the question will turn green (see figure 11 second screenshot) to indicate that evidence documents have been provided for this question. Another option is to add a text reference, the user will be able to type the text he/she would like to add as a text reference. When the text is added, this will also appear in the list of added evidence documents as shown in figure 11.

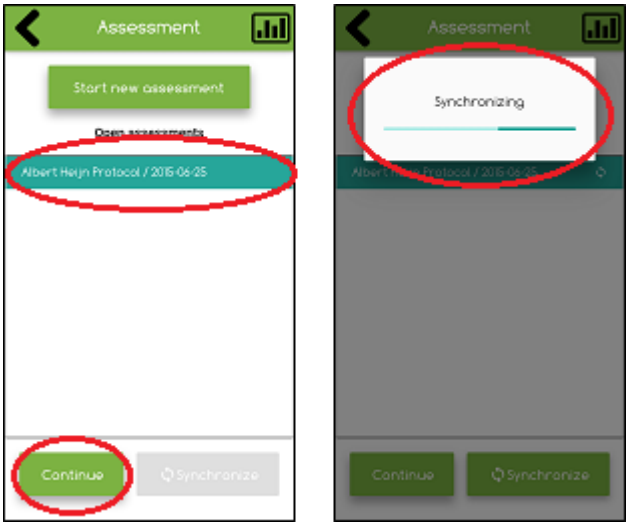

Figure 12. Main screenshot with an open assessment and synchronizing as assessment with the server

The user stops with an assessment when the assessment is finished or when the user returns to the main assessment screen. When returning to the main assessment screen (figure 12), the user will see a list of open assessments, the "continue" button will turn green if one of these are selected. If the assessment is not synchronized, there will be a synchronize icon at the right side of open assessment. When the "synchronize" button is pressed, a pop-up screen will appear showing the synchronization process. When the pop-up disappears that means that synchronization has been completed.

5.2. Changes from initial design

In this section we will show the reasons why changes has been made of the design:

- **An assessment can be started in the crops screen**: In our earlier version an assessment could only be started at the home screen. If the user added a crop they first needed to go back to the home screen and from there go to the assessment screen and start an assessment. One of the growers attending a demonstration of the application said that this would be a useful feature.

- **No option button at the home screen**: We removed the option button, because in our application there is no other option needed than changing the languages. So we simply made a toggle button at the log in screen to be able to switch between English and Dutch.

- **No questionnaires and auto filling of questions:** In the first place this would have been one of the features the client would like to have in their application, but afterwards they said this will be too complex in the amount of time the team had to implement everything.

- **Removed crop comments:** In the earlier version of the application the team implemented this feature, but in one of the demonstration the client told us to remove it because they also removed it from their website.

- **Removed closed assessments:** Closed assessments are not displayed, because the user cannot do any operations with closed assessments. They would not add any value to the application while taking up precious screen space that can be used to list more open assessments.

- **Choose crops screen only has one box with a list of all crops for one organization:** This is because of the size of the mobile phone. If there were two boxes, only three crops could be shown in each box. When the user would like to search in one of the boxes, the user constantly needs to scroll because the boxes are too small. Which will not give the user a good experience.

5.3. Implementation details

5.3.1. Libraries

A major benefit for using standard web technologies to create our application is the ability to choose from a large collection of available libraries. These libraries provide commonly used features and relieves developers from creating these themselves. This greatly reduces the amount of code that have to be written and it lets developers focus on more high level programming problems that make use of these features. Two libraries are being used for the AgriApp, namely jQuery and Materialize CSS.

5.3.1.1. jQuery

jQuery[12] is a fast, small and feature-rich JavaScript. It makes things like manipulating web elements and Ajax requests much simpler. Ajax is used for exchanging data with a server, and updating parts of the web page without reloading the whole page. This simplifies the communication done with the AgriPlace back end which is needed for user authentication and keeping the crops and assessments up to date. It can be said that jQuery is used to provide features for implementing the core of our application rather than appearance, for that we use Materialize CSS.

[12] https://jquery.com/

5.3.1.2. Materalize CSS

The AgriPlace developers reacted positively when the first demonstration was given, but they felt that one library could make the application stand out even more. Andrei, a developer at AgriPlace recommend us to use the Materialize CSS[13] framework for styling the application. We have been using that ever since, because it provides a lot of nice elements that can be used for the GUI. Previously, the GUI was created using standard HTML elements that looked plain and less attractive. A major benefit for using Materialize CSS is that the existing code base does not have to be changed compared to other frameworks that provide styled elements. Elements from Materialize CSS are just pieces of code that can be inserted into the project without requiring any further changes.

5.3.2. Local Storage

The application needs to keep local data about the different users, organizations, crops, open assessments and evidence documents. These data need to persist even if the user closes the application. PhoneGap provides two ways of storing user data, which is Local Storage and an installable plugin for accessing the native Android SQLite database. Local Storage is limited to 5MB whereas SQLite has no explicit size limit. Simple key-value pairs are stored in Local Storage. This is sufficient when there are few user data, but for more data intensive applications like the AgriApp, a SQLite database is the better choice. Databases are more structured and operations for selectively retrieving and changing information are provided. One disadvantage of databases is that these operations are executed asynchronously, this make it harder to code and test the application. The local storage will only be used to keep track of state information when the user navigates through the GUI. A complete diagram of the database is given in figure 4.

[13] http://materializecss.com/

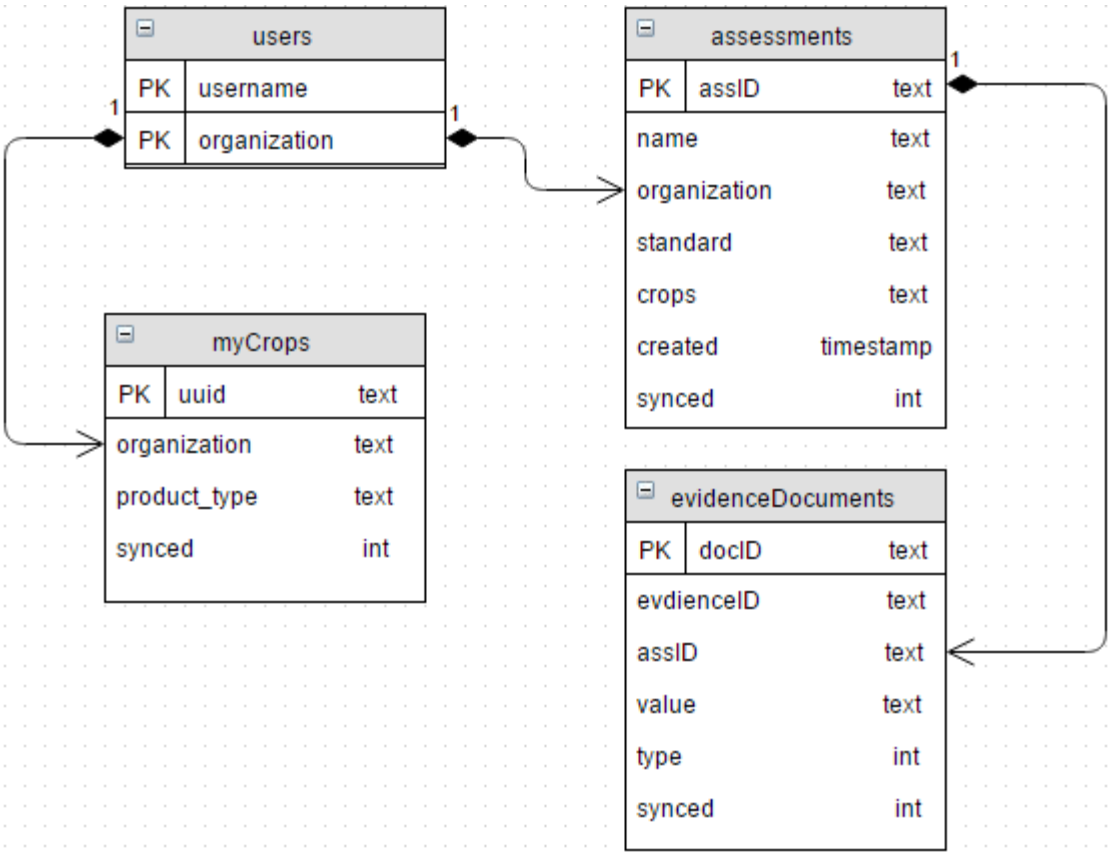

Figure 13. Data model of the SQLite database

In the four tables shown in figure 13 all user data should be stored. The users table contains the different organizations associated with a particular user. For the moment, AgriPlace supports only one organization per user, but this was likely to change in the future. All crops belonging to an organization are stored in myCrops, each crop is given a unique identifier and a “synced” field exists for indicating whether it is an offline or online crop. An unsynced crop means that the crop has been added in the AgriApp, but that synchronization has not yet taken place. Therefore, the server does not know about its existence. This distinction between synced and unsynced crops is made, because if a synced crop is not found on the server anymore, it means that it has been removed. This also why the tables for assessments and evidence documents include a “synced” field. When a user starts an assessment, it is record in the “assessments” table with a timestamp to mark when it was created. Evidence documents can be added for a specific assessment, the included attachments can be of different types: files, photos taken directly with the camera or text references. The value field contains the file name or just text in case it is a text reference.

5.3.3. Synchronization

One of the main benefits of using the AgriApp is the ability to work offline. Hence, data must be stored locally as explained in section 5.3.2. Different copies of the user's data may co-exist, these different datasets should be merged when internet is available and synchronization can take place. Synchronization should take place when the user logs in, so that the user's most current crops, assessments and evidence documents are displayed. There is no doubt that conflicts will occur during the synchronization process. For example, AgriApp users may be working on assessments that have already been deleted on AgriPlace. A decision must be made in how to handle these kinds of situations. After discussing this with the AgriPlace developers, two alternatives came to mind.

The first was to manual lock a certain assessment on AgriPlace before a user can continue with it offline. No further changes can be made to that particular assessment on AgriPlace in order to avoid conflicts. This solution did not sound very appealing, because of the extra step that has to be taken on a computer before the AgriApp can be used to continue with an open assessment. Furthermore, the user may forget to unlock the assessment, preventing access for an excessive long period. To implement this, the AgriPlace back end has to implement the locking mechanism.

A better approach is to let the user perform an assessment at any time and use synchronization rules to handle conflicts. These rules determine which copy of the data should be kept if conflicts occur. The AgriApp is designed as a supporting application for AgriPlace. Therefore, it makes sense to give data existing on AgriPlace higher priority. To simplify synchronization and protecting server data, users are not allowed to delete any data existing on the server. Ideally, the two data sets should be merged where possible, but conflicts are handled by giving server data higher priority. The following synchronization rules are used:

- If the server adds new crops, the app needs to get them.
- If the server deletes crops, the app will delete those crops.
- If the server adds open assessments, the app needs to get them.
- If the server deletes open assessments, the app will delete those assessments
- If the server adds evidence documents, the app needs to get them.
- If the server removes evidence documents, the app will delete those evidence documents.
- If the app adds new crops, the server needs to get them.

- If the app adds open assessments, the server needs to get them.
- If the app adds evidence documents, the server needs to get them.

For implementing these rules, a database with a built in synchronization function would be the easiest solution (Ana, 2014). Couchbase Mobile[14] is such a database solution, but it requires that both client and server use Couchbase for automatic synchronization to work. AgriPlace is currently using PostgreSQL for storing data and it is not an option to migrate the entire database to Couchbase in the current development stage, just to accommodate synchronization. This is the reason why the team has decided to implement the synchronization themselves using regular API calls to the server. More data will be send back and forth between client and server, but it will not be noticeable by the user.

As of writing, the synchronization process has not entirely been implemented. The needed API calls to exchange data with the server are still being tested by the AgriPlace developers. This strong dependency to the server has led to a bottleneck in the development process despite the early requests from the team. When the end points for communicating with the server are ready, the team will complete the synchronization part as is it an important feature of the application.

5.3.4. Architectural pattern

The application has been designed using a Model-View-Controller (MVC) architectural pattern which helps to separate the user interface from other parts of the system (Lethbridge & Laganière, 2005). The model comprises the part that manages the user data and performs correct synchronization. The view renders the appearance of the user interface by using the data in the model. When the user interacts with the systems by, for example, pressing the available buttons on the screen, the corresponding logic that handles these interaction are all contained in the controller module. Making this clear separation of different modules will allow us to adhere to several design principles:

1. Divide and conquer, the three components can be independently designed and maintained.
2. Increase cohesion, functionalities that perform a similar task such as displaying the user interface are now grouped together in the same module
3. Reduce coupling, communication between the three components is minimal and the interface between them is clear.
4. Design for flexibility, the user interface can be very easily replaced or parts changed if necessary.
5. Design for testability, parts of the application can be tested separately from the user interface.

[14] http://www.couchbase.com/nosql-databases/couchbase-mobile

6. Testing

During the project the written code needs to be tested for all sorts of bugs. Some bugs can be found by hand, but such a project also calls for writing your own tests. This section will discuss what kind of tests were performed on the written code and what is not being tested with the methods that we use. For this product we have identified three possible kinds of testing, each with a different level of what is tested. The first is unit testing, which tests if each component performs the functions that it should. Second is behavior driven testing, which mainly concerns about testing the behavior of larger parts of code put together. Lastly we have user testing, where one gives the user tasks to complete with the product and confirms if they can complete them. This can provide useful feedback about usability.

6.1.Unit testing

For this project we have decided not to perform unit testing, because the product mainly consists of web-based elements that are changed and added with JavaScript. Therefore testing the functionality of independent components is not very useful. Not using unit tests means we will not cover some of the code, but the main functionality and the bugs that are caused by those functions can be tested by hand.

6.2. Behavior driven testing

Since most of this application involves making changes to the webpage that runs on the phone and reacting to user input, behavior driven testing is needed to test for those changes and flow in the program. For this purpose we have used the testing framework Jasmine[15] that was provided with the PhoneGap project and Jasmine-jQuery in addition to that, for extra functionalities. We have performed 72 tests on the screens which do not solely use arbitrary functions (such as going to another screen from the home screen). The tests and their results can be seen in figure 14 to 17.

6.3.User testing

For this project we really wanted to do user tests to get some feedback on the design and flow of our application. We have also discussed this with the AgriPlace supervisor. Unfortunately it was not possible to perform user tests, because of the short time span and unavailability of the user demographic.

[15] http://jasmine.github.io/

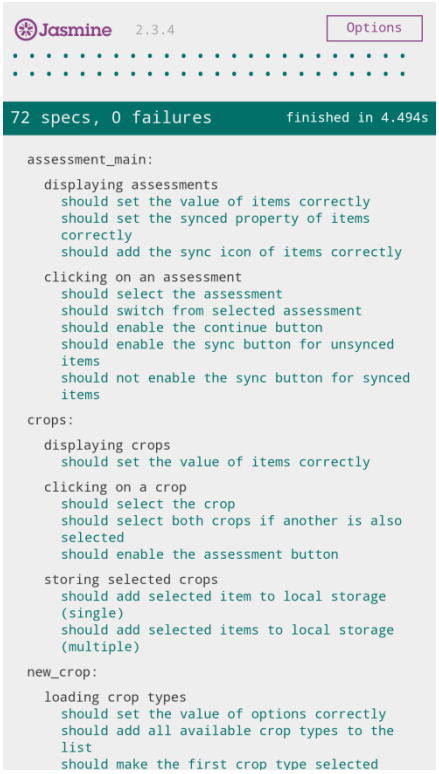

Figure 14. Screenshot of tests (1)

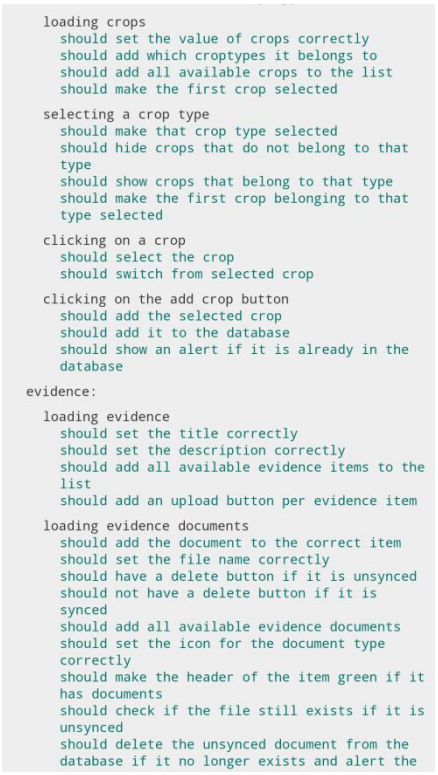

Figure 15. Screenshot of tests (2)

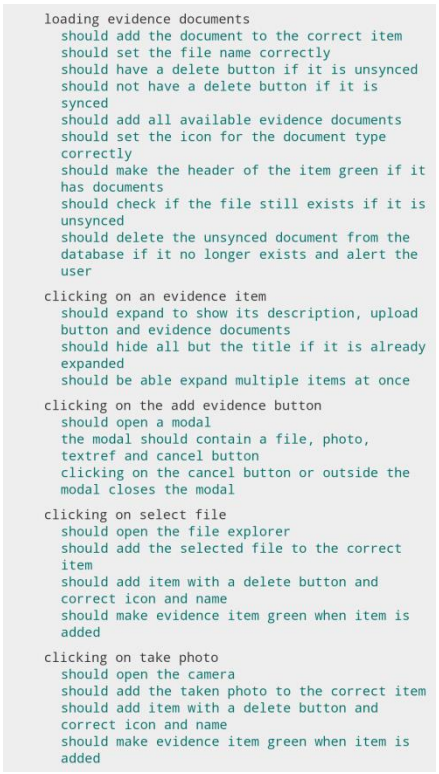

Figure 16. Screenshot of tests (3)

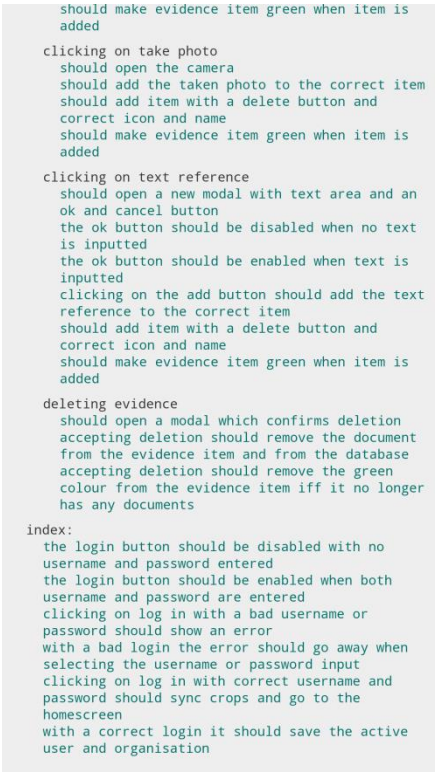

Figure 17. Screenshot of tests (4)

7. Evaluation

Every project brings different challenges that are to be handled accordingly, some are easier to face whereas others are dependent on external factors and thus not within the team's control. With this in mind, an evaluation of the final product is given using the requirements gathered in section 2.5 to see if the project can be called a success. Requirements that are not met should be well supported by argumentation and show that the team has given sufficient effort to solve this. Furthermore, the process will be evaluated in which the team cooperated together and with AgriPlace.

7.1. Final product evaluation

The evaluation of the final product will be made by showing which requirements have been met successfully and those that are not. Ideally, everything that was listed in "Must have" and "Non-functional requirements" should be completed, but unfortunately, this was not the case. An evaluation of these are given in table 1 and table 2. Special attention is given to these unmet requirements to identify the cause and recommendations on how to solve this in the future. Due to the limited time that was available for this project, the items listed in "Should Have" and "Could Have" are excluded. This has been discussed with the client and the project would already be considered successful if everything in "Must have" is met, which is the first milestone of the project and it would offer the greatest value to the users.

Requirements	Met?	explanation
M1. Users can log in using their AgriPlace account	Yes	Users can access their account by providing their AgriPlace credentials when the application starts
M2. Users can log out	Yes	A log out button is present on the home screen which will take the user back to the authentication screen.
M3. Users can start an offline assessment and gather the evidence documents needed for this assessment	Yes	If a user has logged in once successfully, then it has offline access to start an offline assessment.
M4. An open assessment can be resumed offline.	Yes	All started assessments are listed and on resumption the corresponding evidence documents in storage are loaded.
M5. Documents required for an assessment can be added.	Yes	Users can added evidence documents in different ways: browsing in phone's the file system, taking a picture with the camera or by typing a text reference.
M6. Photos required for an assessment	Yes	When selecting the camera option, the user can

can be added or taken directly using the phone's camera application.		select an installed camera application on the phone to take the photo. This will be stored in the phone's memory.
M7. An open assessment can be stored locally on the smartphone.	Yes	All data regarding an assessment are stored in a local database.
M8. A locally stored assessment can be synchronized manually when internet connection is available.	no	This has not yet been implemented, because several end points with the AgriPlace servers have not been provided to the team. This makes communication and thus synchronization impossible. As soon as these end points are available, synchronization will be implemented.
M9. Crops can be added.	Yes	The user can add a new crop by selecting from a list of all crops known by AgriPlace.
M10. The list of crops can be viewed.	Yes	All crops belonging to a specific organization are listed.
M11. It is visible which offline assessments have been synchronized.	Yes	Assessments that have been synchronized are listed without a synchronization icon next to it, also, the button for manual synchronization will be disabled.

Table 1. Evaluation of the must have requirements including explanation

Requirements	Met?	explanation
NF1. The application must have an intuitive interface which is easy to use and understand without any training	Yes	The GUI has been kept very simple and basic. Buttons are large and self-explanatory. The flow of the app is similar to the AgriPlace website
NF2. The code must be of sufficient quality and provide documentation so that it can be reused by other programmers.	Yes	The SIG evaluation of our code in Appendix C provides the necessary feedback to make sure it meets industry standards. Since the first evaluation, the quality increased from three to four stars and this can be even further improved by incorporating the feedback from the second evaluation. Also, enough comments are provided and this final report will clarify the synchronization process and data model.
NF3. Navigating through the interface must feel responsive and smooth.	Yes	As expected, choosing PhoneGap for cross-platform development has not impacted the user experience negatively. The screens and user data are loaded in an acceptable short period, short enough not to make the user lose focus.
NF4. The user interface should give the same look and feel as the website.	Yes	The same color has been used in the app for the buttons and header as can be found on the AgriPlace website. For consistency, the same icons are used to indicate certain operations.

		Also, both Agriplace and AgriApp are implemented using web technologies, this adds to the feeling of browsing through a website.
NF5. Users should follow the same steps taken to perform a task on the website as much as possible.	Yes	An assessment is started in the same way as on the website. Users select a standard and the corresponding crops which will take them to the screen for providing evidence documents. Though an extra shortcut in the app to quickly start an assessment through the crops screen has been introduced.
NF6. The application must be able to run on Android OS and possible iOS.	No	The goal was first to create a working prototype for Android including synchronization. No time was available to support iOS as well, but the development of the application was done with eventually porting it to iOS in mind. That is why a Cross-platform approach was chosen which allows a large part of the code to be reused for the iOS version.
NF7. Communication between the server and the smartphone is encrypted.	Yes	Communication with the server is done using the encrypted https-protocol.

Table 2. Evaluation of non-functional requirements including explanation

Two important goals have not been reached, these are the synchronization process and support for iOS. The latter goal was not required for the first prototype, but synchronization was a must have feature. Because of this, it is fair to say that the project as a whole was not successful. The team has defined the required end points needed for the synchronization very early in the research phase, but the result is still in testing. Nevertheless, code for synchronization has already been written with the assumption of a working end point, but the correctness cannot yet be verified. It is expected that the team can finish this last part of the project in relatively short time since the code is already written and the synchronization rules well defined in section 4.3.3, it only needs testing and probably debugging. Despite this, all other features work as expected and the several demonstration by the client and farmers were received enthusiastically.

7.2. Process evaluation

The chosen software methodology method as described in section 2.4 has worked very well. The team got used to the rhythm of having a working prototype ready at the end of each sprint. Having the same sprint duration of two weeks as the AgriPlace developers helped to keep both parties up to date with the latest development changes. By working in their office in Amsterdam at least once a week, we maintained contact with the company without having to make a long trip every day. Not much guidance

was required from the AgriPlace developers despite us being inexperienced with web technologies, all information was available on the internet. Still, in Amsterdam, there was the opportunity to just sit down and brainstorm with the developers about certain ideas. On other days, communicating through Skype for short questions went just as easily. In retrospect, the company showed a very kind and willing attitude towards us. Enough workspace was provided and they were prepared to answer questions at any time. The only problem that has occurred was the delay in creating the necessary end points for implementing the synchronization. This led to a bottleneck in the development process and as a result, the project goal was not met.

In the initial planning, managing the crops should be made fully functional before the development of the assessments part begins, because a user must be able to select or add new crops to include in an assessment. Due to the missing end points for retrieving and uploading an organization's crops, the team started developing the front end for the assessments while awaiting the requested end points. When the entire front end was finished, the team moved more extensive testing and writing the final report up the agenda.

Apart from the Wednesdays spent in Amsterdam, the team mostly worked on the TU Delft campus. Sometimes it was hard to find adequate work space without being send away, because no room is officially reserved for this project. Nevertheless, working in Delft most of the time has allowed the team to get more work done. There were days that everybody would work at home independently and it seemed tempting to do that more often, but we would not recommend that in the early stages of development. In the beginning of the implementation phase, lots of decisions have to be made that require consent from all team members. This is why almost all work is done in Delft to keep the communication lines short. To divide the work, a list of remaining tasks was created and the team would decide which tasks had highest priority. These tasks were divided among the team members.

8. Conclusions

AgriPlace wanted to make their service accessible on a mobile phone. Growers are often outside and there is limited to no internet connectivity. Even in these situations, it would be nice if the grower could do assessments and add evidence documents. That is when the developed AgriApp will provide the solution.

Growers with a smartphone can now do assessments anywhere in the world. The camera on the phone is an excellent tool to take photos of physical evidence documents or crops, these can then be added to an assessment. This user created data can be synchronized with the server when internet connection is available so that the changes made are also visible on AgriPlace.

Almost all requirements for this project are met, except the synchronization part. This was the most difficult part to implement, not only because of the technical aspects, but due to the dependency on the AgriPlace developers for providing the required end points. We expect to implement this feature very soon, because the process is already defined and code has been written. It only needs some testing and debugging. Still, the team is satisfied with the final product as it is now. Apart from the synchronization, all other features work very well and several people from AgriPlace and local growers have reacted positively about the application. The project is seen as a success.

9. Recommendations

The created prototype was meant to serve as a basis for future development by the AgriPlace developers. There are several recommendations that can be given to them. First, in order to optimize the synchronization, a migration from the current server database to a couchbase database would reduce the amount of data that has to be sent back and forth between server and client. Another solution could be to implement the server side synchronization themselves, this should return a change log of data that has occurred after a certain timestamp. Second, we would highly recommend to create documentation for the back end of AgriPlace. It was very hard to gain a proper understanding of the system, because there was no documentation available. This makes the company very dependent on the current developers. And finally, more features can be implemented in the AgriApp that are also found on AgriPlace. People are tending to work more often on mobile phones and it would be a valuable asset to AgriPlace if the user has the flexibility to access all features anywhere in the world.

Appendix A: Project description

Consumers have increasing expectations about food. Food needs to be save and sustainable beyond the standards set by national laws. The number of voluntary standards has been increasing in response, covering food safety (e.g. GlobalG.A.P.), environmental (e.g. Rainforest Alliance) and social issues (e.g. Fair Trade). These standards prescribe a range of policies and practices that should be in place in order to receive certification. For growers this brings two main challenges: increasing and unclear buyer requirements and a significant overlap in information requests. These challenges impede growers in developing countries to access more rewarding markets.

To lower the compliance burden of farmers AgriPlace has developed a compliance service. Here is how it works. On AgriPlace, growers find all standards in one place, with clear guidelines and easy-to-use online registrations. These registrations are 'compliant-by-design', which means that upon filling out the online registrations a large chunk of the standard control points are already answered, saving grower's time and confusion. Registrations can be re-used across standards. When selecting a new standard, previously filled-in registrations for other standards are preloaded, marginalizing overlap between standards. Historical registrations only need to be updated when renewing certification the next year. The grower is owner of its data and with a few clicks shares his data with whoever he wants: advisors, auditors, buyers, etc. By providing auditors insight in the data registrations before the audit, audit time and costs may be reduced.

We aim to roll out the compliance service to different countries the coming years, including developing countries. An (high speed) internet connection is not always available in rural areas; neither in developing nor developed countries. At the same time the adoption of mobile devices is increasing. Therefore the goal of this project is: to develop a mobile app for the AgriPlace Compliance Service with offline functionalities.

Some very high level requirements, in need of further specification in iterations with the project team, include: 1) Usability. The mobile application should ideally be usable for users with limited computer literacy. This also implies that the mobile information flow shall differ from the current web-based version. 2) Capabilities to store content and user data locally, synchronize content and user data with server 3) Scalability. Solution should be scalable to different devices (phone, tablet) and different platforms (Android first, iOS, Windows). E.g. explore the use of hybrid solutions like PhoneGap. 4) Maintainability. The mobile app should be easy to keep in sync with the web based application.

Appendix B: Backlog user stories

ID	AS A USER, I WOULD LIKE TO ...	SO I ...	ACCEPTANCE	PRIORITY	HOURS
1	have a log in screen	can access my account with a username and password	When the application starts the user will see the log in screen	1	4
2	not get in the account if the username and/or the corresponding password is incorrect	will not allow anyone who has no username with the corresponding password in to an account	Access only granted when a username with the corresponding password is correct	2	8
3	See 3 buttons (Crops, Assessments, Log out) on the home screen	can reach all the functionality of my application	Buttons for the different functionalities are visible	3	4
4	be able to log out	can Log out of my account	When the user sees the log in screen again after he logged out of his/her account	4	4
5	see a list of my crops	can see which crops are added to the list	When the button is pressed of crops, the user will see the list of crops	5	4
6	add crops to the list of crops I have put online	can add crop(s) into the list	The list of crops is increased with the crop which the user wants to add	6	4
7	see a list of open assessment(s)	know which assessment(s) still need to be completed	After the assessment button of the home screen is pressed, the list of open assessment(s) will be displayed	7	8
9	see a button to start a new assessment	can begin with filling in a new assessment	After the assessment button of the home screen is pressed, the button of start a new assessment will be displayed	8	8
10	see all standards	can request a new assessment for crop(s)	After pressing the start a new assessment button, the list of button(s) of standard(s) will be displayed	9	8
14	have a list of crops when I have selected a standard	can select the crops for an assessment	When crops can be selected and the user can go to next screen of the assessment	10	4

			regarding the selected crops		
30	see all a list of all the evidence documents required for the selected standards	easily the documentation necessary	a) sorted by compliance level b) then sorted alphabetically	11	8
11	add a file or multiple files (PDF, picture, etc.) to an evidence type	provide the requested document(s)	Reference to the corresponding documents has been made and the screen shows which file has been uploaded	12	8
12	see the question that I need to answer	can answer the questions correctly	Each question of the assessment is displayed on screen one by one	13	4
13	see an area to fill in my answer	can answer my questions	There are buttons and textbox on each question screen	14	4
23	have a "back" button when answering a question	can go back to the previous question	User can see the "back" button on the question screen and the User can go to the previous question	15	4
24	have a "next" button when answering a question	can go to the next question	User can see the "next" button on the question screen and the User can go to the next question	16	4
25	have a "return" button when answering a question	can go back to the assessment screen	User can see the "return" button on the question screen and the User can return to the assessment screen	17	4
15	locally save a filled in assessment	can continue filling in my assessment later or synchronize	When continuing an open assessment, locally stored answers for that assessment are filled in.	18	20
16	have a button to synchronize the locally saved assessment(s)	can manually synchronize the assessment(s) when internet connection in available	When a button for synchronizing is displayed at the main assessments screen and locally saved data is synchronized with the server	19	24
31	the evidence types for which files have been uploaded, to auto answer corresponding question in the assessment questionnaire	can save time	a) Reference to the corresponding documents has been made and the screen shows which file has been uploaded b) CRUD	20	16
26	have a "Go to" button when answering a question	can jump to a specific question	User can see the "Go to" button on the question screen. When button pressed, there will be a pop-up	21	4

27	have a dropdown box at the pop-up screen	can select a chapter in assessment to jump to	User can see dropdown box and is able to select a chapter in the dropbox	22	2
28	have a "cancel" button at the pop-up screen	can continue filling the current question	User can see the "cancel" button. When the button is pressed, the pop-up will disappear	23	4
29	have a "Go" button at the pop-up screen	can jump to the question that is filled in the textbox in the pop-up	User can see the "Go" button. When the button is pressed, jump to the requested question	24	4
18	see my profile	can see if all information is correct	profile details are visible	25	4
19	to edit my profile	I can update my profile	User can make changes to profile details	26	4
20	see my certificates	know which certificates I have achieved	show all achieved certificates	27	4
21	add a certificate	can show to other users which certificates I have achieved	able to add new certificate	28	4
22	see a confirmation when a modification is completed	know that the changes are saved correctly at the mobile phone and/or Server	the user is always notified about a change in user data, the text "saved" is shown	30	2
8	see a list of closed assessment(s)	know which assessment(s) I don't need to fill in anymore for a certain crop	After the assessment button of the home screen is pressed, the list of closed assessment(s) will be displayed	31	6
17	have some question(s) filled automatically when there is an overlap with an older assessment	don't need to answer every question again	All questions with overlap are filled in automatically	32	16

Appendix C: SIG evaluation

SIG first feedback:

[Aanbevelingen]

De code van het systeem scoort 3 sterren op ons onderhoudbaarheidsmodel, wat betekent dat de code gemiddeld onderhoudbaar is. De hoogste score is niet behaald door een lagere score voor Unit Size en Duplicatie.
Voor Unit Size wordt er gekeken naar het percentage code dat bovengemiddeld lang is. Het opsplitsen van dit soort units in kleinere stukken zorgt ervoor dat elk onderdeel makkelijker te begrijpen, te testen en daardoor eenvoudiger te onderhouden wordt. Langere units in dit systeem zijn te vinden in de html code, zie bijvoorbeeld de bestand 'evidence.html'. Het is aan te raden kritisch te kijken naar de html code binnen dit systeem en deze waar mogelijk op te splitsen.
Voor Duplicatie wordt er gekeken naar het percentage van de code welke redundant is, oftewel de code die meerdere keren in het systeem voorkomt en in principe verwijderd zou kunnen worden. Vanuit het oogpunt van onderhoudbaarheid is het wenselijk om een laag percentage redundantie te hebben omdat aanpassingen aan deze stukken code doorgaans op meerdere plaatsen moet gebeuren. Een goed voorbeeld van duplicatie in dit systeem is te vinden in de bestanden common.css en new_crop.css. Het is aan te raden om dit soort duplicaten op te sporen en te verwijderen.
Als laatste nog de opmerking dat er geen test-code is gevonden in de code-upload. Het is sterk aan te raden om in ieder geval voor de belangrijkste delen van de functionaliteit automatische tests gedefinieerd te hebben om ervoor te zorgen dat eventuele aanpassingen niet voor ongewenst gedrag zorgen.

SIG second feedback:

[Hermeting]

In de tweede upload zien we dat de volume van het systeem is gestegen en de onderhoudbaarheid een stuk is verbeterd. Jullie zitten nu op 4 sterren.

Bij Duplicatie zien wij een kleine verbetering vooral in de css code. Jullie kunnen echter verder de score van Duplicatie verbeteren als jullie proberen meer kritisch de html bestanden na te kijken, waar nog steeds duplicatie te vinden is. Zie bijvoorbeeld de bestanden crops.html en new_crop.html.

De score van Unit Size is gelijk gebleven. Lange units zijn nog steeds te vinden in de html code. Jullie kunnen deze score verbeteren als jullie proberen deze code waar mogelijk op te splitsen.

Tot slot is het goed om te zien dat jullie testcode hebben geschreven.

Op basis van deze observaties kunnen we concluderen dat de aanbevelingen van de vorige evaluatie gedeeltelijk zijn meegenomen in het ontwikkeltraject.

Appendix D: Info sheet

General Information:

Titel of the project: AgriPlace Mobile App

Name of the client organization: AgriPlace

Date of the final presentation: July 9, 2015

Description:

Growers all over the world face cumbersome administrative work when they want to certify their products by doing assessments. AgriPlace created a platform to make certification easy, but it is only accessible through a web browser. Growers often live in areas where there is limited internet connectivity. So the client has asked for a mobile application that enables the growers to do the assessments offline and perform synchronization of the data with the AgriPlace server when there is internet connectivity.

The most challenging part of this project was the synchronization. Besides the difficulty of keeping the data on the phone and server consistent, it was challenging to manage the delay of the requested end points. These were necessary for allowing the application to communicate with the server. As of writing, some end points, but not all, have been provided and the team is prepared to implement the synchronization of assessments and evidence documents in the coming days before the final presentation.

The research was mainly focused on the best way make the application available to users on different phone platforms like Android and iOS, a cross platform-development framework called PhoneGap is used.

We have chosen SCRUM as software development methodology with sprints of two weeks. Our client also had sprints for two weeks and we attended each other's demonstrations to detect changes in requirements and planning.

The final product is an Android application that serves as a prototype for future development. The application is able to add crops, do assessments and save these data offline, only the synchronization part is still not functional. Testing is done using behavior driven testing, this makes sure that certain actions produce the desired result on a higher level.

The application was designed with flexibility and maintainability in mind and can be easily be ported van Android to other mobile platforms. We recommend introducing more features that are found in the web version of AgriPlace and a more efficient method of synchronization.

Team Roles:

Andy Chiu
Contributions: Synchronization, Data Storage, Final Report

Ka Ho Man
Contributions: Front End, Final Report

Julian Faber
Contributions: Front End, Testing

Maarten Den Uijl	AgriPlace Product Owner	maarten.denuijl@agriplace.com
Tomas Klos	Algorithmics Group at TU-Delft	t.b.klos@tudelft.nl
Andy Chiu	Team member	A.S.Y.Chiu@student.tudelft.nl
Ka Ho Man	Team member	K.H.Man@student.tudelft.nl
Julian Faber	Team member	J.K.Faber@student.tudelft.nl

Bibliography

Ana, B. (2014, July 17). *Simple server-client syncing for mobile apps using Couchbase Mobile*. Retrieved June 10, 2015, from https://www.infinum.co/the-capsized-eight/articles/server-client-syncing-for-mobile-apps-using-couchbase-mobile

Charland, A., & Nitobi, B. L. (2011). Mobile Application Development: Web vs. Native. *Mobile Computing, 9*(4), 49-53.

Cross-Platform Framework Comparison: Xamarin vs Titanium vs PhoneGap. (n.d.). Retrieved May 20, 2015, from http://www.optimusinfo.com/cross-platform-framework-comparison-xamarin-vs-titanium-vs-phonegap/

Freibauer et al. (2011). *Sustainable food consumption and production in a resource-contrained world.* European Commission – Standing Committee on Agricultural Research (SCAR) .

Gong, J., & Tarasewich, P. (2004). Guidelines For Handheld Mobile Device Interface Design. *In Proceedings of the 2004 DSI Annual Meeting.*

Lethbridge, T. C., & Laganière, R. (2005). *Object-Orientated Software Enigneering: Practical Software Development using UML and Java* (2nd ed.). McGraw-Hill Education.

Sommer, A., & Krusche, S. (2013). Evaluation of cross-platform frameworks for mobile. *Lecture Notes in Business Information Processing, 140*, 120-138.