The Development of an Accessible Web-based Quantum Transportation Device for Older Users

| **Stephanie Lee, Thomas Butterwith, Kurtis Mulgrew, Ron Schoenberg**  **Final Year Industrial Project**  **BSc (Hons) Applied Computing**  **University of Dundee, 2014**  **Prof. John Arnott**  Converted to a PDF file. Together with complete software documentation and a user manual. |
| --- |

***Abstract* –** The potato plant is the fourth most grown crop in the world and therefor a significant portion of the world’s food production and feeding basis for cattle. There are more than one hundred known diseases and pests which threaten the potato plant and have had dramatic consequences in the past. The James Hutton Institute in Dundee, Scotland, has asked us to design a diagnostics application for smartphones which enables an unexperienced and/or unskilled user to identify pests and diseases in both the tuber, leaves and stems of a potato plant with appropriate guidance and troubleshooting upon identification. The project case study focuses on farmers in Malawi. The application is therefore aimed to help those farmers specifically. This includes catering to internet access standards, smartphone types and language barriers. The finished product fulfils all the essential requirements and enables non-technical access to the application database as well as an intuitive interface and concise help upon identification of a potato plant problem.

# Introduction

The potato plant has played a central role in the world’s nutrition ever since it has been exported from the South American Andes and is the 4th most important food crop after Rice, Wheat and Maize [1]. The potato plant tuber, the edible portion of the plant, is very perishable in comparison to other mass-nourishing products minimising the export and trade potential of potato tubers. This limits the uses for the tuber to direct consumption, livestock feeding and starch production. Regardless of the intention for potato farming, the potato is susceptible to a wide range of diseases which can have a dramatic impact on the yield of the farming and has had impacts on entire nations such as Ireland in 1740-1741 [2]. While the knowledge and use of pesticides and the knowledge about diseases have increased ever since the Irish Potato Famine, it is still crucial to educate farmers and provide easy identification tools, especially to farmers in developing countries who face the same diseases but lack access to pesticides and expert knowledge.

This project aims at creating an initial diagnosis tool for potato diseases and pests for farmers in developing countries who have no easy access to Microbiological diagnosis tools, the internet, pesticides or professional on-site help.

# Background

Current analytical tools for disease and disorder identification are the LFD field test, the FTA card test, two smartphone applications for site testing, and regular textbooks.

The LFD test works similar to a pregnancy test in that it provides a quick way to test a leaf sample for a single trait of a disease. The advantages of this test are the instant feedback, the quick application, the low cost and the versatility as it can be carried and handled in the field. The drawbacks of the LFD test are, that a single test only tests for a single trait and there are hundreds of potentially devastating diseases. This could mislead a farmer to classifying a disease as a harmless anomaly due to an irrelevant, falsely conducted or misinterpreted test.

The FTA card test consists of a cardboard envelope onto which the farmer presses up to four samples of leaves or stems. The cardboard is pre-treated to conserve the genetic matter for up to a decade, allowing the card to be sent to an institute with an appropriate DNA tester. The farmer would then receive an information package about the disease or pest with guidance on how to handle the problem. The benefits of the FTA card are, that the card is small and therefor easy to ship worldwide as well as cheap to produce. The DNA sample on the card can be thoroughly analysed to identify any traits of any known disease and the result of the test will be as exact as the current technology and research levels allow. The drawbacks are the time it takes for the feedback to arrive, as it could take weeks for the delivery of the card from remote areas, and extended times to analyse the sample. Also, the plant sample has to be squished onto the card, making the application non-intuitive and prone for contamination if applied in the field. The FTA card test is therefore the most precise test, but also the test which requires the highest concentration from the applicant as it may otherwise be contaminated or provide an insufficient data set to analyse.

A quicker alternative to the FTA card is an on-site evaluation of the plant, tuber or pest with the help of a smartphone application. The International Plant Nutrition Institute (IPNI) offers an application which allows the identification of diseases in multiple crops [3], but only helps identify lacks of nutrients, and does not give suggestions to pests or diseases. The currently only alternative is "Potato Pests" by Leah Tsror, an Apple Application which lists and gives information about Diseases [4]. The benefits of this application are, that diseases are described accurately and symptoms are described in a detailed manner. Drawbacks are, that the user is expected to know the name of the requested disease prior to using the application and that no advice is given to the user upon successful identification. The application is therefore not a useful first-diagnosis tool, but rather useful in combination with the FTA card, when the name of the disease is known and further education on the symptoms is desired.

More common techniques for evaluation are textbooks and field guides. While there are several editions in multiple languages, the range of examples in each book is fixed and updates in research require the purchase of a new edition. A textbook will hardly fit into the farmers’ trouser pockets, making them largely unavailable and unpractical in the field.

It can therefore be concluded that a smartphone application is a useful format for a field diagnosis as smartphones are widely abundant in first world countries and surprisingly common in developing countries. Even the matter of internet access is not the unimaginable task it seems to be in developing countries, as most people will have access to the internet in public places, despite a common lack of electricity at home. An application can therefore be kept up to date with updates and the existing phone is given an additional task with an application for disease and pest identification.

# Specification

## Problem Specification

The problem identified is, that there is no quick way for an uneducated farmer to analyse a potato disease or pest in the field, as all the existing techniques require either external input or expert knowledge of disease. The product in design may therefore not rely on external input, except for acquisition and updates, and may not rely on the assumption of expert knowledge or further input.

## Problem Specification Explanation

The problem specification was concluded upon on the basis of Prof. Lesley Torrance’ description of the economic, social and technological situation in Malawi, Africa and Europe. Based on the focus on Malawi for this project, it was concluded that certain features such as offline access and Android OS compatibility had to be prioritized to enable correct functionality.

## Requirements Gathering

Following the agile method requires close customer contact allowing quick interaction and continuous feedback. The representative for the James Hutton Institute for this project was Professor Lesley Torrance, who was available through email correspondence and weekly meetings.

The ideal agile scenario involves meetings with the client (representative) repeatedly to establish user stories and acceptance tests. Due to the time constraints, this requirements-gathering phase was not conducted in a dedicated meeting, but through note taking during conversations about requirements. An example for this would be the requirement that the application should be scalable for future purposes, which emerged from discussions with Professor Torrance about the envisioned product.

After a group evaluation phase, in which we analysed the feasibility of the requirements, we came back to Professor Torrance with our requirements list. We reiterated the requirements and explained the planned product, which would fulfil the important requirements. An important step in this process was to clearly show rejected requirements such as speech and visual recognition, as they would not have been feasible within the project timespan.

The second phase of user interaction was the acceptance test for the product. While we did not receive clearly formulated acceptance statements, we did have a good idea of the customers’ expectations of the product.

The primary client interaction happened during the weekly meeting, which allowed us to demo the prototype and first functioning version of the app respectively. We were able to gage feedback for the website and application and get an impression of the customer reaction to the product. The acceptance tests were therefor hands-on testing of the product rather than verbal statements for each user story.

## Initial Work Schedule

The timescale of the project is a 3 week period in which the entire research, planning, creation, testing and evaluation cycle will be completed. As an agile approach was selected for the development of the project, production started early within the first week as a means of concept and feasibility testing. Key cornerstones of the project scope were the design of the database, the limitations of the XML protocol and the application implementation of the dynamic XML parsing GUI builder. The initial goal was to have the shell for the website, the database and the application done for the midpoint of the sprint duration to grant sufficient time for optimization, debugging and thorough testing.

The only mandatory deliverable for the project is a smartphone application which allows an on-site diagnosis of a disease or pest. Due to the scalability requirement, we have concluded that a minimum of two deliverables are needed: The smartphone application and a website. The websites’ purpose is to allow data entry and manipulation and is therefore the backbone of the application.

The normal student allocation for the Industrial project aims at having teams of five computing students, with some teams of six. Due to a no-show and an unexpected graduation, we started off with six people, but ended up with a four member group. This enabled us to split into programmer pairs and tackle tasks in teams. The first week was spent in pairs creating the website and database as well as the application and xml parsing respectively. The second week was focused on combining the two projects and getting the core functionality running, while the third week was dedicated to design optimization, debugging, testing and project management. The key milestones of the project were the final deadline on Friday the 3rd of October, and the half-way meeting with the client on Wednesday the 24th of September. While the objective for the final presentation was clear, we aimed at having a first running version ready for the midpoint meeting in order to gage feedback and evaluate present design concepts with the customer.

While the work in programming pairs was clearly divided along the margin of website and application throughout the first week, the pairs split and regrouped for the remaining time as the products had to be combined and project management required independent work on the backlog and report. The last week involved optimization from all members on specific portions of the product as well as input for the report and the presentation.

# Design

## Design Considerations

We designed the application on a cooperative design basis, assuming that the user and we, the designers, are on a similar level of experience when interacting with the application and website. The presumption was made on the basis that we, as computing students, know very little about potato plants that exceeds the purchased store product. It is that same basis of knowledge which we assume the farmer in Malawi, who may not be aware of the different kinds of diseases, but rather just aware of an ill plant, could possibly have. There is a high probability that the average farmer knows a lot more about the diseases of the potato than we do, but we assumed that no knowledge is present to be able to cater to those that need the advice from the application the most.

For the web interface, we assume that the data entry and editing will be performed by computing professionals with similar knowledge of databases as we have, meaning we can tolerate drawbacks in design if it benefits the efficient manipulation of the database. This thesis is backed by the customer who ensured that all data manipulation will be done by computing professionals.

We settled for the development of an Android application due to the availability of cheap android devices as oppose to highly expensive devices from Apple. It was discussed to provide a windows Phone implementation as well due to the involvement of Microsoft in Developing countries through offering windows surfaces and phones for promotional prices. The decision to not develop a Windows Phone application was made on the basis of time constraints and the desire to produce a reliable and thoroughly tested product rather than two potentially buggy and untested products. From our customer information, it is also known that android phones are widely spread and popular amongst the population of Malawi. This means the application can cater to a large user base instantly without requiring the purchase of a new phone.

## The Data

We decided to use a database which holds the disease and symptom information as oppose to a text file or other data structure which would not allow the same extents of scalability as a database would. The database type chosen is a SQL type, as concurring database types such as Cassandra cater to large data amounts, and only excel in efficiency if used on very large data sets, which this application will never reach, even if all possible future extensions are considered.

The SQL database has a total of seven tables, one for the problems, i.e. Diseases and pests, one for the symptoms, one to link the latter two, two for the images (problem and symptom), one for the user data for the website and one for the versions. The database design allows for easy entry of new data and uncomplicated altering of existing data. The most important quality of the database is easy querying of the data and conversion of the content to XML code, which can then be downloaded to the application from the application website.

The phone will run a slim version of SQL, namely SQLite. The data will be downloaded by the phone in a XML format and then reconstruct the database locally from the XML string.

A typical SQL database allows unplanned queries through the linking of tables, enabling alternative uses for the database in the future as opposed to singular use through access-restrictive database types such as Cassandra.

## The Website

The websites purpose is to enable the administrators of the application to add, edit and delete data from the database. The website is not intended for regular customer/user use and does not display data in an intuitive or easily readable format. This could be changed in later versions of the website, but is not currently implemented.

The current website features a login feature designed to protect the content from unauthorized access as well as ensuring integrity of the data through limiting the editing rights to trained staff. The core content of the website is a menu from which the user choses whether to add new data or edit / delete existing data. Once the changes are saved, the database will add a timestamp to the changed content flagging it for the next update whenever a user checks for version updates for the application. The design of the website is currently not aimed at a wide crowd of editors and contributors, as it is table based and does not provide an intuitive user interface. The customer statement to the matter is, that only one or two professionals will edit the application content, and therefore use the website, meaning that upon training, these professionals can edit content correctly and likely more efficiently than through a complex user interface which would essentially provide the same functionality.

## The Application

The purpose of the application is to allow the user to navigate to a potato disorder or disease easily, given only a couple of symptoms visible to an amateur. While the easiest option would have been to provide a list of all the problems and have the user chose the relevant one, it would have been hard to handle with more than the initial 20 problems and therefore not easily scalable. We settled on the idea of a decision tree which is modelled around the problem symptoms. The user starts out on the home screen of the application and is prompted with three categories such as "Pests", "Leaf and Stem Symptoms" and "Tuber Symptoms". With the sample size of diseases ideally reduced to one third, the user is prompted with the next layer of granularity such as "is the leaf crinkled or spotted" or the colour of the symptom. Each question reduces the available number of diseases and eventually guides the user to a narrow selection from which to choose a disease. The greatest design challenge for this stage was to accommodate for multiple different symptoms which are all traits of the same disease, meaning that the database must accommodate for multiple symptoms for each disease, namely a linking table allowing multiple relations for single diseases.

A side effect of the multiple-symptom feature is, that a problem could be found under multiple categories of symptoms, which is why the sample size of problems to choose from is initially increased. The final design aims to allow a diagnosis using multiple, possibly very different, symptoms.

At the end of the decision tree, on the finest level of granularity, which will leave only a couple of options to choose from, there will be a link to a detail page for a specific disease/ pest or problem. This detail page features all the data given from Prof. Lesley Torrance with hyperlinks to further information. The information pages feature a design which aims to deliver data in the easiest possible fashion and avoids unnecessary confusion by first presenting descriptive and informative text for the disease followed by example pictures of symptoms which the user can compare to the example in hand.

The first application design created was a wireframe prototype of the android application layout (Figure 1), which was revised, but quickly agreed upon as the best basic design.

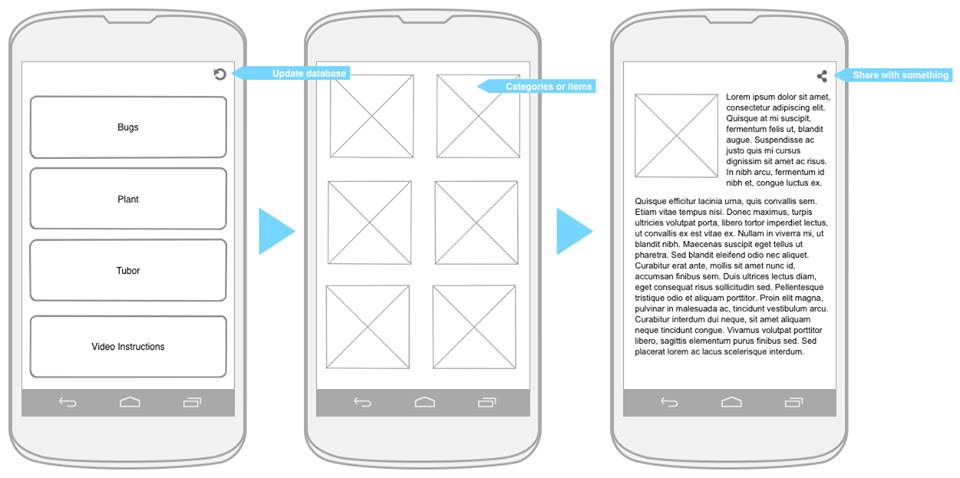


Figure 1. The Initial Application Wireframe Design

The only notable difference to the final application design is the significance of the menu screen (Figure 1 middle screen). This screen will be replicated and customized according to the database dynamically.

The main difficulty in the application design is the dynamically changeable structure of the symptom-driven decision tree menu. Since the structure of the tree could change with every update of the application database, it is important to have a menu page design which accepts one or many options as opposed to a fixed amount of links. Future extensions might involve an overview of the remaining diseases and pests within the selection, as the current design requires the user to navigate to the end of the decision tree regardless of the number of remaining options

## The Update Function

For the future development of the app, it is vital that the app can be updated to keep up with current research and provide the best analytical options for a quick field diagnosis. For the purpose of updating the application contents, a timestamp has been added to all data in the database which is updated every time that data is created or changed. The user is given an update button on the home screen of the application through which the timestamp of the users’ last update is compared with the data in the database. Every item in the database which has a later timestamp than the users’ timestamp is downloaded to the user application. The user application is then recompiled, creating the (possibly) new structure of the decision tree to incorporate the changes of the update. While the downloading of the files and the recompilation of the app structure take a lot of time, it is by far the most efficient way to update the application, as downloading the entire structure every single time a change is made would take up much more time and resources and be potentially unnecessary, as changes might not change the layout of the decision tree at all. It is also incredibly inefficient to delete all images in the application, just too then download all images again, even if no image was changed, or just a few changes were made.

# Implementation and Testing

## Implementation

The group split up into programming pairs throughout the first week based on ability and task allocation. The second and third team were approached as a team again in order to combine the work of the two pairs and enable concentrated work on focused areas such as database optimization, CSS, revisions of the application and management tasks.

Kurtis and Thomas paired up to design and develop the Database, Website and XML functionality while Stephanie and Ron shared the development of the Application, the XML parser, Ethics Applications and Project Management tasks.

Figures 2 and 3 show the first functional version of the website without CSS or specific design developments after the first week:

Figure 2. Initial Content overview page

Figure 3. Initial Content Manipulation page

Figures 4 and 5 show the finished and functional website with an appropriate design and layout for data manipulation:

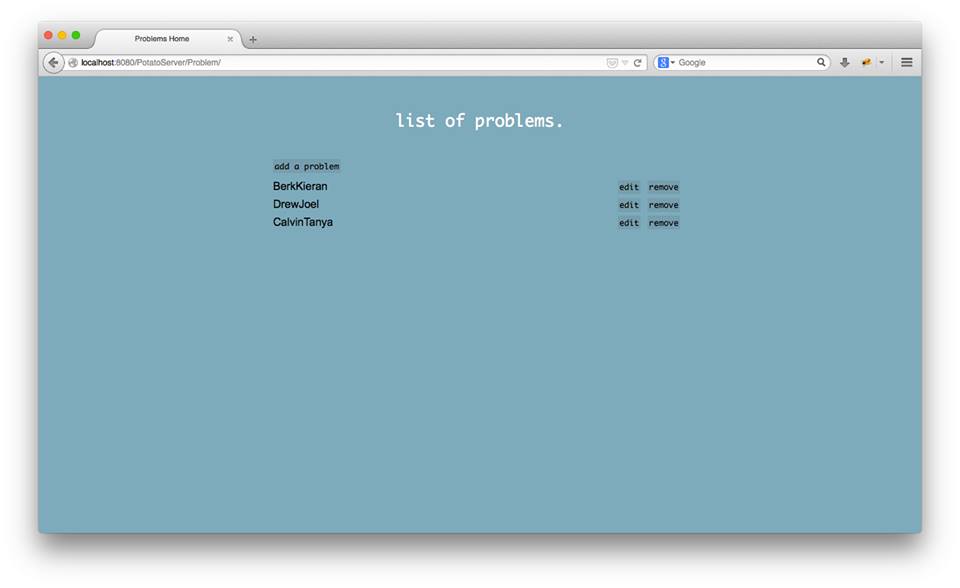
Figure 4. Content page with final design

Figure 5. Final Content Manipulation Page

Stephanie and Ron paired up throughout the first week to work on the first version of the application. The application design was only finalized throughout the third week of the project as the first two weeks were spent on the XML parser and the SQLite database. Figure 6 shows the final application title page with the initial navigation options and the update button. This page also links to two web based videos for the use of the FTA card test and the LFD test.

Figure 6. Application Title Page

Figure 7 shows an example of a navigation page as it would be found when navigating from the title page to a subsection and symptom. These pages are created dynamically after an update is completed. Therefore the number and content of the links may change.

Figure 7. Sample application symptom navigation page

Figure 8 shows an example of a Problem content page. The content page holds a description of the disease or pest, other symptoms, if applicable, and suggestions for the user in order to cope with or get rid of the problem. This page also shows a series of pictures for the problem to serve as a visual aid and facilitate diagnosis.

Figure 8. Application Problem Page

## Debugging

The first functional version of the product was finished in the middle of the second week, so at the halftime of the project duration, leaving sufficient time to add data and test the functionality thoroughly. We found several minor bugs in the application during the beta testing phase such as invisible or hidden menus which were meant to be visible, text exceeding the previsioned space for the text and hyperlinks which weren't in the correct and usable format. The only greater bug found is, that if the update is not completed, the timestamp is not going to be updated partially, as the data is not added in a sorted manner. This means that, if the user has a connection difficulty or the update terminates for any other reason, the downloaded data is ignored when the user tries to update the application in the future. A solution for this problem would be to sort the data to be downloaded by timestamp and keep track of the latest downloaded timestamp. The user application would then obtain that last downloaded timestamp and could resume where the download terminated. Due to time constraints we did not include this functionality in the first version of the application, it could however be done easily in future sprint iterations.

## Testing

For the user tests, four casual potato farmers were found to test and evaluate the application. It was concluded that a non-professional, casual potato farmer would be ideal, as advanced experience and expert knowledge is not a perquisite for the anticipated user group of the application. Given that the application is aimed at unskilled farmers who lack knowledge about the diseases their potato have, we resorted to testing the application with subjects who had very limited knowledge about farming.

For the application testing, we finished a working version of the application with 20 diseases and pests. In order to be able to obtain quantitative as well as qualitative data, we devised two tests for the user.

Test number 1 consisted of two runs in which the Subject is given a written description with images of a potato plant with Virus Y and Late blight with the aim to find the correct disease definition. During this test we were able to both observe the users interaction with the application as well as record the time the user took to reach the correct disease and the number of incorrect navigation attempts and false diagnosis.

The following Table 1, shows the results of the test runs:

Table 1. Results from Subject Interactions

|  |  |  |  |
| --- | --- | --- | --- |
| **User # / Run # / Disease #** | **Time for completed navigation (sec. +- 0,5s)** | **Number of incorrect navigations** | **Number of false diagnosis** |
| 1/1/1 | 20 | 4 | 1 |
| 1/2/2 | 11 | 2 | 1 |
| 2/1/2 | 18 | 3 | 2 |
| 2/2/1 | 7 | 2 | 1 |
| 3/1/1 | 18 | 4 | 2 |
| 3/2/2 | 9 | 1 | 0 |
| 4/1/2 | 22 | 4 | 2 |
| 4/2/1 | 13 | 1 | 1 |

Disease 1: Virus Y

Disease 2: Late blight

The data shows the evolution from the first to the second run for each user. It is clear that the users had some confusion during the first run and a couple of false diagnosis. The second run was, on average, completed much faster and the number of false navigations and diagnosis more than halved. In order to minimize experimental error, we switched the two disease to be found. This means that users 1 and 3 had to search for Virus Y first, while users 2 and 4 had to search for Late blight first.

The qualitative feedback received was fundamentally positive, and the users agreed that the navigation could probably not be more intuitive than the current version. Some concerns were voiced on the subject of scalability, as the app could potentially grow to incorporate a very complex menu. There were however no concise ideas as to how that complexity could be reduced in the future.

Test number 2 was a trial and error run where the user was given the application and told to browse and play with it. While the test followed no structured procedure, it is surprisingly helpful for design purposes to watch a user naturally use an application. Findings of this test were, that the menu buttons could be a bit larger by default, in order to enable larger font sizes on small devices and that scroll menus are not obviously labelled as such, resulting in the presumption by some users that there was no lower section to a page. A problem which two of the testers encountered was the back-function of the phone, which is a built in button on the bottom right of the phone, and not within the application screen. Based on this test, we increased the size of buttons and labels and included a back button within the menu structure.

Through this user testing, largely performed with student peers who have grown potatoes, we found that the application concept is fundamentally sound, as it allows the navigation to certain diseases from symptoms only, as required by the customer.

A suggestion we got from one test user was, that the granularity is often very exact, but also very tedious. So while there might only be one choice of a submenu within a menu, it still has to be opened to reach the disease page. This means that the user has to potentially navigate multiple symptom pages with only one option on each page, leading to a single disease. The solution we thought up for this problem is to show all remaining results in a subsection below the menu options, to enable a direct skipping to the disease as oppose to forcing the user to navigate to the bottom of the menu-decision-tree. For time constraints, we did not implement this feature though, as it also meant to add an entry of each disease to each page of the parent symptom tree during the dynamic creation of the decision tree. The final product design would then be, that all diseases or all pests are shown when taping the "diseases" or "pests" button, and each sub question reduces the number of options left. This design may prove cluttered in the future if the application is scaled to several hundred diseases, but could again be cleaned up with a limiter reducing the number of displayed results.

# Evaluation

## Usability

Usability testing with the user group was performed exclusively on the application, as the target audience for the website is a computing professional. The usability analysis for the website was created in cooperation with Kurtis, who inserted the bulk of the preliminary data.

### The Application

The application aims to offer a quick diagnosis tool for potato diseases and pests, and user tests have shown that, after a short introduction and learning phase, the user can quickly navigate to the disease or pests on a potato plant given only a picture and short description of the problem. Therefore the ease of learning for the application is ideal, given it took the subjects less than one minute to understand and work with the application. While we produced a video which serves as a step-for-step guide to the application, we think it is absolutely possible to learn the functioning of the application from trial and error.

We concluded that the design fulfils the customer criteria in offering a symptom based search, and we were unable to suggest a more intuitive way of navigating to a disease or pest given a plant sample or sample picture. The design of the menu is held as slick as possible to avoid distracting the user with options or animations.

We believe that the efficiency of use of the menu is both on a very high level and open for optimization as the menu structure can be changed within the database at any time, and even cases of multiple symptoms for the same pest or disease are taken care of in an intuitive way. The efficiency the use can still be further enhanced through the use of colour coding of colour symptom options. This could enhance recognisability of symptoms and facilitate the use of the application further.

We are certain that a farmer, regardless of experience, will appreciate the presence of the application as a diagnosis tool. While a given farmer may know most of the common diseases from experience, it is likely a welcome option to be able to quickly consult an application in the case of an anomaly or to obtain further guidance on an issue. This memorability of the application distinguishes our application from other venders and is likely to reserve it a permanent space on the farmers’ android device.

The error frequency for the application is relatively high at the moment, given an average of 1 false navigation during the second run of each test subject. The test subjects were usually immediately aware of the false diagnostic through the pictures given in the information page, which could potentially be further reduced through the use of several images within the symptom menus. This means that although the error frequency might be relatively high, the user can always use the back button to return to the navigation menu through which a different symptom-path can be chosen. The assumption for error frequency is, that if there was an average of one error on a sample size of 20 diseases, then there will likely be 5 errors on a sample size of 100 diseases. We concluded, that an easily identifiable false-diagnosis, even several, are tolerable given the easy use and the help provided upon correct diagnosis. No system will be able to provide a correct diagnosis on the first try with the same accessibility, making our application useful for a user and helpful in the field in comparison.

The subjective satisfaction of the users was positive, with the basic consent being that the application could probably be very useful to the farmers in developing countries who lack access to professional help and technology or even just textbook resources.

### The Website

The greatest unexpected complication when working with the website was the insertion of data into the database. While the data was provided entirely by the James Hutton Institute, the insertion proved difficult due to the creation of the symptom-based-decision-tree, which had to be created within the symptoms table of the database. This task was only accomplished after we laid out the different diseases and symptoms and created a physical decision tree after which we could model the database. This will likely be a consistent problem for future additions to the database, as the website interface does not currently provide a graphical representation of the data. The solution to this complication could be to offer a graphical representation of the decision tree on the website allowing the insertion of nodes and menus graphically, or to provide a digital map of the decision tree as a visual aid. While the former could be implemented in the future, the latter proved to work well during the insertion of the primary data. A third possible solution for the addition of data to the decision tree, is a smartphone application which allows the insertion of menus into the app directly, creates the xml representation of the decision tree from the menu structure, and updates the database accordingly. It was deemed too complicated for this sprint though, as the web site already exists and could be altered easily to facilitate edits.

We concluded that the simple adding, editing and deleting functions of the website were as intuitive as possible. The menus guide the user directly to the desired options and avoid unnecessary confusion.

## Other criteria

### Group Analysis

The group of four members worked collaboratively on almost all aspects of the project. The requirements and requirements evaluation phase were completed as a team, after which programming pairs were formed and tasks were allocated based on ability and availability. In order to facilitate task allocation, a sprint backlog [Appendix 6] was created and tasks were distributed accordingly.

The group attendance was excellent, and while a strict daily attendance was neither necessary nor required, the attendance for the scrum meetings was spotless and the meetings productive and concise. There was equal input from all members and queries were addressed instantly.

Some effort was made throughout the first two weeks of the project, to establish contact to Chavez, the fifth group member. Professor John Arnot however discouraged further search and insisted we concentrate on the project rather than what could possibly be an allocation mistake.

Possibly more group work analysis

### Efficiency Analysis

The most complex efficiency feature within the product is the use of a decision tree to enable the user to filter irrelevant diseases and pests leaving only a short list for the user to choose from. The relationship between the number of diseases and pests is logarithmic, therefore the number of menus and submenus will only increase slightly when the app is scaled to the anticipated 100 entries.

The problem with the application updates was a lot more challenging to solve. The basic solution would have been to publish an update for the application and prompt the user to download the entire application at once and reinstall it. This would have meant to download a potential amount of ~600 images which, even at reduced size, would have meant high traffic volumes and long waiting times for people without Fibre optic connections. This would have been especially annoying if the update was a minor one.

The solution was a partial update, through which only the new or changed data is downloaded to the application. This means, that the user will still have to download the entire data set with the first acquisition of the application, but any further updates will only have to download the changes made to the dataset since the last successful update.

# Summary and Conclusions

The final application offers a quick tool for on-site field diagnosis of pests and diseases in potato plants and tubers. The design is optimized to allow quick navigation and updates of the application to ensure a valid and up-to-date dataset. The application runs on all devices with the latest or version-related Android Operating System (OS) versions, meaning it will be able to run on most existing devices without reconfigurations and the devices themselves will be of the cheapest category thanks to the open source nature of the Android OS.

While the app works correctly under lab circumstances, the main point of criticism is the update functionality. While it is very sophisticated as it is, any error during the connection will currently require a completely new download of the data. Also, if the user first acquires the application, it will be empty. The user has to update the data when first running the application, meaning that a potential cold start and initial use in the field will be deemed unsuccessful as the data will not be present in the store version.

The greatest success in this group project was the group management. While there was an official group leader, decisions were generally made by the group as opposed to single individuals or a single leader. All tasks were successfully distributed among the members and through the use of daily scrum meetings [Appendix 4: Meetings Minutes] and a comprehensive sprint backlog [Appendix 6], it was always clear what members were working on and what the next step in the project was. The greatest lesson learned for all members of the group was, that with a thorough group management, based on group member abilities, a project as large as this one can be broken into manageable pieces.

The group concluded that the only possible point of improvement for future projects was, that the first week was not used to its full extent and that the last week was thus not as structured as it could have been. User testing had to be done on the afternoon of the final Wednesday, leaving little time for project management optimization and presentation preparation. Other than that, the group project was very successful and the group worked together closely.

## Acknowledgments

The group would like to thank the James Hutton Institute, and Prof. Lesley Torrance for the Images and Information provided as well as the instant feedback received during the development process.

# References

[1] "International Year of the Potato 2008 – The potato" (PDF). United Nations Food and Agricultural Organisation. 2009. Retrieved 26 October 2011.

[2] Gavin, Philip. "Irish Potato Famine." The History Place, 12 June 2000. Web. 23 Sept. 2014.

[3] International Plant Nutrition Institute Apps. *Crop Nutrient Deficiency Photo Library App*. Computer software. *Crop Nutrient Deficiency Photo Library*. Vers. 1.1. International Plant Nutrition Institute, 9 July 2012. Web. 22 Sept. 2014.

[4] Tsror, Leah. *Potato Pests*. Computer software. *Leah Tsror, Ph.D. - ARO*. Vers. 1.1. Apple / ITunes, n.d. Web. 17 Aug. 2012.

# Appendices

The following Appendices are to be found in the Appendix directory on the CD:

1. Website Source Code
2. Database Creation SQL Code
3. Application Source Code
4. Meeting Minutes
5. Project Log
6. Sprint Backlog + Burndown chart