The Development of a Test Script Management Solution which Leverages the Advantages of Technology and Human Marking

Victor Soudien – SDNVIC001

Supervised By: Gary Stewart

Category		Min	Max	Chosen
1	Requirement Analysis and Design	0	20	15
2	Theoretical Analysis	0	25	0
3	Experiment Design and Execution	0	20	5
4	System Development and Implementation	0	15	15
5	Results, Findings and Conclusion	10	20	15
6	Aim Formation and Background Work	10	15	10
7	Quality of Report Writing and Presentation	10		10
8	Adherence to Project Proposal and Quality of Deliverables	10		10
9	Overall General Project Evaluation	0	10	0
Total Marks		80	80	

Department of Computer Science

University of Cape Town

2014

Abstract

Despite the increasing prevalence of technology in education, the process of test script management has remained largely unchanged. Research in the area shows work done on creating fully automated solutions but these have found limited success due to the limitations that they place on examiners. Automated solutions also have difficulty managing any ambiguity on the test scripts which is traditionally managed by markers through discussion with each other.

This research aims to determine whether or not creating a system which combines elements of human marking and technology could improve the current test management procedures. It details the design, implementation and evaluation of such a system which includes a mobile tablet based marking interface.

This report also discusses the significant results of using this system in a real world scenario and possible improvements which could be made.

Table of Contents

List of Figures

List of Tables

1. Introduction

The management of test scripts at tertiary institutions involves their collection; distribution to markers; evaluation and redistribution to students. The current methods in place rely on the physical test scripts and require a considerable amount of time and coordination between multiple individuals. These methods increase the waiting time for students who ideally need to get feedback as soon as possible in order to improve on future assessments. Current test management also lacks the collection of any data about the tests aside from the student's final mark which leads to a lack of historic data for examiners to reference when creating new tests.

Advancements in this fields often focus on the creation of completely automated electronic solutions. While these have found limited success, their greatest weakness is in their inability to handle ambiguity either in the student's answer or the examiner's questions. Other solutions such as those which utilize Optic Mark Recognition (OMR) require institutions to change their current procedures to accommodate the method by forcing examiners to use a predefined format for the test or limiting the types of questions they can ask.

Through the development of an electronic test management system which combines elements of technology and human marking, this research wishes to answer the following questions:

- Does a system which combines technology and human marking increase marking speed?
- Is a tablet and stylus interface considered intuitive by markers?
- Is a tablet interface considered more intuitive when compared to a web interface?

1.1. Project Overview

ScriptView was developed to aid in answering these questions. It is an electronic test management system which aims to enhance the current test management procedures by leveraging the advantages of technology at specific stages in the process. These advantages include the rapid processing of large amounts of data, automated error checking and the ability to efficiently maintain a digital paper trail. It is a complete test management solution which aims to reduce the time taken to mark and redistribute tests as well as capture and analyze the results.

The system allows users to scan the paper versions of tests and mark them on either a web or tablet interface. The scanned tests are automatically sorted and stored on a central server according to the course and test name. This server is access controlled in order to ensure the security and integrity of the tests.

As mentioned, the scripts can be marked on either a tablet or web interface once they are available on the server. Each of these interfaces is optimized to offer the most intuitive marking experience given their limitations. These limitations include, screen size and input type. Both interfaces aim to decrease the time that is required to mark a test while maintaining a high level of accuracy.

ScriptView allows for the distribution of marked scripts to the students and for a summary of the marks, of a particular test, to be sent to a Teaching Assistant (TA). The format of the email sent to students allows them to quickly assess in which areas they lost marks and thus where they need to improve. The summary sent to the TA is in Comma-Separated Values (CSV) format which is compatible with most learning management systems and thus reduces the time needed to capture these marks on another system.

The diagram in Figure 1 shows the various components within the ScriptView system.

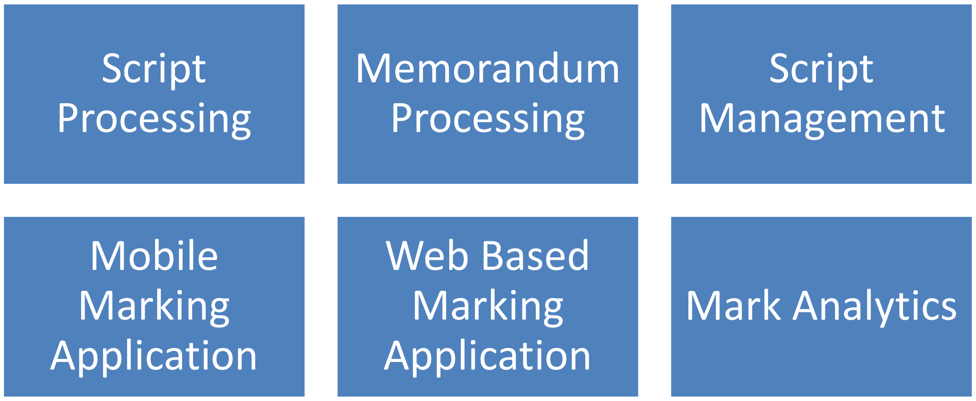


Figure 1: Components of ScriptView

1.2. Report Format

This report details the design, implementation and evaluation of the mobile marking application, script processing and memorandum processing components.

2. Background

This section presents the relevant findings from previous and current work that either directly or indirectly influenced the design and development of the system.

2.1. The Current Test Management Procedure

The information presented here was gathered through observation of the current procedures. These observations were made at The University of Cape Town (UCT) during the researcher's time as an academic tutor.

Students write paper based tests which are collected by the TA. The TA then distributes these tests equally among the tutors. Each tutor is responsible for marking the entire test and writing the total marks awarded on the first page of the script. The tutors gather at a single location at a given time to mark the scripts so that any ambiguities can be discussed and clarified.

Once all the tests have been marked, they are returned to the TA who captures the final marks onto the university's learning management system. The marked tests are placed at a collection point within the department building and the students are notified via email that their scripts are available for collection.

An important observation was that students often neglect to collect their scripts or the scripts are misplaced. This entire process from the point of collecting the test scripts until they are available to the students can take up to five days. However, it should be noted that this is because the UCT Computer Science department puts an emphasis on quickly returning scripts to students. In general the turnaround time is much longer.

2.2. Functionally Similar Software

The marking of paper-based tests has remained largely unchanged however, there has been significant work done on trying to digitize the marking process which shows some promising results.

One of the most widely recognized techniques is that of OMR which utilizes optical equipment to interpret marks on a paper [1]. These marks are usually indicated in a grid format with cells corresponding to letters or digits. Commercial products which implement OMR are available and have had limited success. One of the key constraints of OMR is that it limits the examiner in terms of what type of questions they can ask. This is because OMR is only suited to multiple choice which implies that examiners can only ask questions up to the applying level in Bloom's taxonomy (See section 2.3).

An alternative to this would be to use a system that could analyze free form answers. This kind of technique has been explored, especially in the case of essay marking.

Researchers such as Christie [2] have found success applying automatic marking to the marking of essays. The researcher describes a technique used to mark both style and content which requires acceptable style metrics to be set up before marking, as well as manually marking a certain amount of scripts to calibrate the algorithm. Such a system would allow for questions from all levels of Bloom's taxonomy and would therefore be more widely applicable.

Pulman [3] did similar work in that he attempted to use machine learning techniques to automatically mark short free form answers. Although he draws no conclusions on how easy his method is to customize to different questions, there is a problem which is identifiable in both his and Christie's approaches. That problem, is training data. A system that is more generally applicable to a wide range of tests is preferred since time does not have to be spent on customizing it for each new type of test.

There are also certain issues which could arise which Thomas [4] expanded upon in his work. Thomas performed an experiment in which he compared the results of a fully automated marking system and a manual marking system. The tests intended for automatic marking were taken on a computer and the other tests were done on paper. The researcher noted that certain problems could arise during the taking of the test that could only be solved effectively by human markers.

The most problematic of these being that of ambiguity, either in the student's answer or the examiner's question. Students may also not be able to express themselves clearly, especially if they are taking the test in a language other than their first language. Both of these problems could be overcome by manual markers who used their knowledge of the domain and discussion with fellow markers to assign appropriate marks.

2.3. Forms of Assessment

An advantage of electronically processing tests is that it creates the opportunity for the collection of large amounts of data about students' progress. This data can then be analyzed to predict learning patterns and activities that might indicate a risk of failure [5]. As one of the main goals of the system is to allow for the recording of such data, it was essential to understand what kind of information an examiner may look for and how it would be used.

It was found that tests are often divided into two distinct categories namely formative and summative assessment. Formative assessment is used during the course of a particular section such that the results can be used to inform how the student or examiner should progress [6]. Summative assessment on the other hand is used at the end of a section to gauge the extent of the student's learning for grading, certification or the evaluation of the effectiveness of the teaching method [6].

The terms formative and summative assessment however, do not describe the format of the tests but rather their function. As a result, a range of question types can be used in both. A classification of these types, based on their cognitive complexity, was developed in 1956 by B. Bloom and revised (see Figure 2) in 2001 by L. Anderson [7]. The cognitive level of complexity increases as one ascends from remembering to creating. The taxonomy provides a framework which examiners can use to ensure that they are testing all aspects of a student's knowledge and allows them to create tests which assess the most appropriate levels for the given situation.

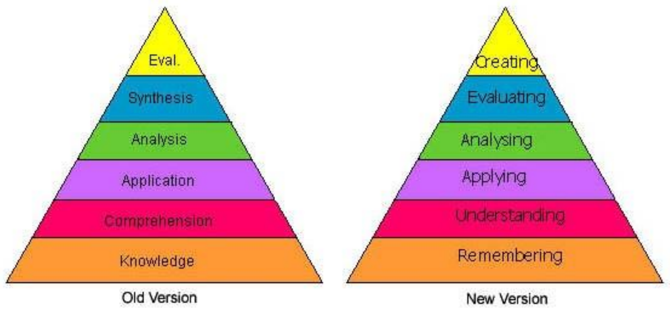


Figure 2: Original and revised Bloom's taxonomy [7]

Aboulsoud [8] claims that formative assessment is more valuable to students since it provides them with feedback which they can use to improve their future work thus increasing their chances of academic success. He suggests that for formative assessment to have this effect, it should be followed by immediate feedback which clearly indicates areas in which the students should improve while acknowledging their effort to achieve the objectives of the assessment.

This implies that any system developed to aid the marking process should allow for rapid feedback, with the characteristics outlined by Adoulsoud, as well as allow for the marking of questions at various levels of Bloom's taxonomy.

2.4. Converting Physical Tests to Digital Format

The biggest bottle neck in the processing of scripts, for electronic solutions, appears to be the scanning of test scripts so that they can be processed and stored electronically. This was noted by Doctor H. Suleman (personal communication, 24 April 2014) of The University of Cape Town who has implemented a system which involves the scanning of marked test scripts to automate the capturing of marks on the university's learning management system known as Vula. The system he developed only requires that the cover page of the test be processed since it is the one which contains the marks and student information. However, all pages are scanned since they are emailed to the student as feedback. Due to this, the quality of the scanned document needed to be high enough so that the student would still be able to read the feedback written by the marker and that the image processing algorithms could successfully detect the student number on the cover page which was indicated by shading certain pre-defined areas similar to what is used with OMR.

To accomplish this the documents were initially scanned at 300dpi, but it was decided that the scanning was too slow and thus proved infeasible. To improve the speed of scanning, the resolution was changed to 200dpi and the documents were scanned in black and white. This sufficiently reduced the scanning time while maintaining the readability of the document.

A problem that was not addressed by Dr. Suleman during our interview, was the size of the scanned documents. This was however addressed by Doctor J. Tangkuampien (personal communication, 1 May 2014), who also scanned documents at 300dpi and had to reduce it to 200dpi. He however, reduced the resolution due to concerns about the file size and not the scanning speed. When scanning at 300dpi the file size was approximately 4MB and decreasing the resolution halved this size. Another difference between his scanning solution and that used by Dr. Suleman is that he scanned the documents in colour. While scanning the documents in black and white would have further reduced the file size, Dr. Tangkuampien noticed that doing so interfered with the visibility of the text, if students used highlighters. This is because when scanning in black and white, very light highlighters such as yellow do not show and dark ones obscure the text.

Dr. Suleman used a combination of OMR and manual name entry to name scanned files whereas Dr. Tangkuampien used a fully manual approach. With the first approach a special cover page needed to be created whereas with the second the existing test format could be used.

S. Chetty (personal communication, 5 May 2014) described a method which uses optical character recognition (OCR) to name files. This method required that a certain area of the script be reserved for the student number, which eliminated the need for a dedicated cover page as the area for the student number could be included on the same page as the first question of the test. This is essential as he suggests that a large part of improving the test management process will be a compromise between the existing process and the new electronic solution. By eliminating the creation of a cover page the overall process can remain as simple as possible.

Minimizing file size while maintaining the readability of the final document is essential to the final system as limited storage is available; scripts for multiple tests need to be stored and image processing needs to be performed on the documents.

2.5. Software Interaction

In order to make the best use of the tablet and stylus interface, it was necessary to understand what users have found intuitive in the past.

Alisi [9] discusses the concept of natural interaction during the description of systems implemented to improve the experience of museum visitors. The researcher suggests that natural interaction can "reduce the gap between computing and ordinary physical things" however, it would require that the interfaces differ from traditional human-computer interaction such as the use of menus and icons.

The Point At system described by Alisi allows the user to point at a character in a picture in which they are interested and the system will provide them with more information on that character. Since this is a natural action and similar to how the user would traditionally ask a tour guide for information, it allows the technology to become a transparent medium instead of overwhelming the experience.

This point is further expanded upon by Malizia [10] who believes that a natural interface, especially gesture based ones, should allow users to interact with the software using the same gestures they would use with the actual physical object. The researcher states that users should not have to learn an arbitrary set of gestures to use the software but instead the software should allow gestures which take the user's habits, background and cultural aspects into consideration. The researcher recognizes that this goal might not be achievable but knowledge of these aspects means that interfaces that are as natural as possible can be designed by carefully considering what will be intuitive for the target audience.

In order to provide students with useful feedback, Dr. Tangkuampien suggests that a three-pane view is the most useful. The three-pane view describes the screen and document layout used that presents the viewer with the question, student answer and model answer in a single view. During his research Dr. Tangkuampien has found that this view has proven the most beneficial to both the marker and the student. It allows the students to easily assess what they did incorrectly and how they can fix it. The software shown during the interview maintained the traditional view of a test script, this is as a set of multiple pages exactly as they were scanned with the model answer added to the right column

Dr. Suleman observed that when working with electronic test scripts one does not have to maintain this view. He suggests that it is possible and may be beneficial to tailor the view of the test script to the device on which it will be marked. When changing the view however, the ability to add feedback should still be maintained as Fowles [11] suggests that this is both reassuring to the marker but is also essential if the marking is challenged in the future.

2.6. Summary

Considerable work has been done in the area of fully automated marking, yet it has found limited success due to the constraints it places on the overall process. While techniques such as OMR, restrict the examiners in terms of the types of questions they can ask. In order to be effective a test management system should aim to improve on the current procedures and not replace it. Any interfaces created for this system should aim to mimic the manual marking techniques as closely as possible in order to improve user acceptance.

3. Design and Implementation

3.1. Software Development Methodology

A software development methodology is a framework which guides the planning, implementation and evaluation of a software system [12]. The two most well-known frameworks are the waterfall and agile methodologies. While there is variation within these two categories, the central theme of each variation remains the same.

The waterfall methodology places an emphasis on planning and performing all tasks in a sequential manner as can be seen in Figure 3. The idea is that by performing thorough planning at the beginning of the project, changes can be avoided at later stages. There is also a strong emphasis on having only a single implementation phase in which the entire project is developed. The advantages of this approach are that the planning and resulting documentation can often be beneficial to inexperienced teams and it allows for the strict monitoring of the progress of the project.

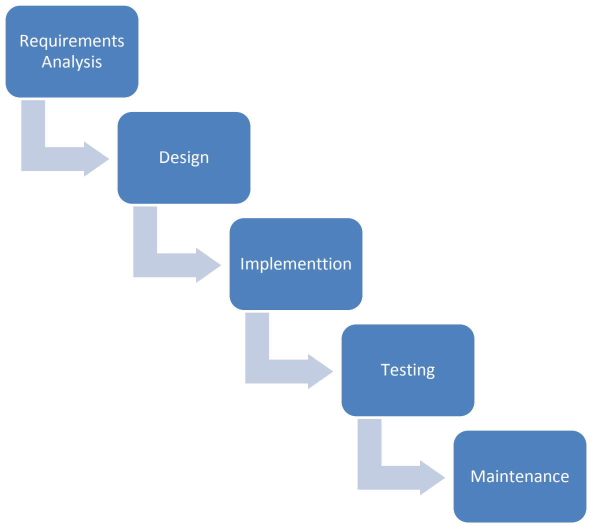


Figure 3: An overview of the waterfall methodology

Where the waterfall approach aims to mitigate having to incorporate change at later stages of the project through planning, the agile methodology embraces change. An overview of the agile methodology can be seen in Figure 4. If one considers that the 'discover' phase shown is equivalent to the requirements analysis phase of the waterfall approach, then it is clear that the agile methodology consists of multiple iterations of the waterfall methodology excluding the maintenance phase. Each of these iterations is known as a sprint and the result of each is commonly a deployable portion of the final system. The advantages of this approach are that it allows developers to respond to changing user requirements - as long as these remain within the scope of the system; it promotes the involvement of users within the process thus ensuring that the final product better meets their expectations and it allows for the creation of user testable products throughout the development process.

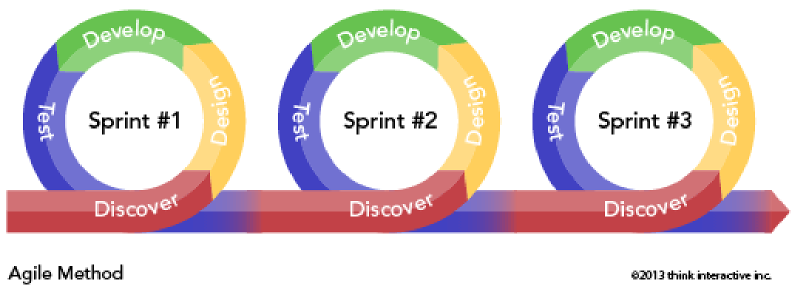


Figure 4: An overview of the agile methodology

Since ScriptView is intended to enhance a current process, the feedback of users involved in the current operations is essential. This meant that it would be necessary to create versions of the final product which could be used for user testing throughout development. It was thus decided that following the agile development methodology would be the most beneficial.

3.2. Programming Paradigm

Throughout development an Object Oriented (OO) approach to system implementation was followed. The OO programming paradigm states that components should be thought of as objects with an emphasis on their properties, behavior and interaction with other objects [13]. This fits the ScriptView system well as each of the various components may be written in different languages but will need to communicate in order to function correctly.

Even though this communication will be necessary, all objects will be created with the aim of having low coupling and high cohesion. Coupling describes how dependent one object is on another and cohesion describes how related elements within an object are [14]. By having low coupling and high cohesion, within the system, it will be more reliable and maintainable [15].

3.3. Evaluation Design

Since ScriptView is intended to replace an existing process it was decided that a user centric approach to system evaluation will be followed. This approach allowed for the input of users to have an impact on the final system and thus improve user acceptance.

3.3.1. Data Collection

The effectiveness of the system was evaluated by using seven tutors from the CSC1010H Computer Science course. Each tutor was required to mark three test scripts. One was marked as they normally would, without the system, while the other two were split between the two interfaces.

The marking took place in a controlled environment in the sense that outside interference was limited to only the course convener, TA or examiner. The tutors marked all tests in the same location and it was a location which was familiar to them.

Upon completing both sets of tests the participants were asked to complete a survey, which can be found in Appendix A. Based on the feedback changes were made to the system before another test was conducted under these same conditions. After which another survey was completed.

The results from these two surveys is what was analyzed in order to answer the research questions.

3.3.2. Data Analysis

Most responses were given using a Likert-scale so it was appropriate to use mode or median [16] to draw conclusions about the data. The mode and the second most frequently occurring value was used to determine how divided the responses were. This was because if the mode was 4 and the second most frequent was 1, then it meant that tutors either strongly agreed or strongly disagreed and therefore the reason behind this had to be further investigated.

3.4. Process Design

ScriptView is intended to enhance the existing test management procedures as described in section 2.1. This means that the system needed to be designed in such a way that it would not disrupt the current operations to a large degree.

Figure 5 shows the overall process flow of ScriptView. This flow was inspired by the existing test management systems but decreases the amount of human intervention required. The components in green indicate the parts which were completed by the author, Victor Soudien, while those in blue were completed by another researcher, Zahraa Mathews. Although this report will only focus on the components completed by the author, this section provides an overview of the entire process in order to provide more context for the discussed components.

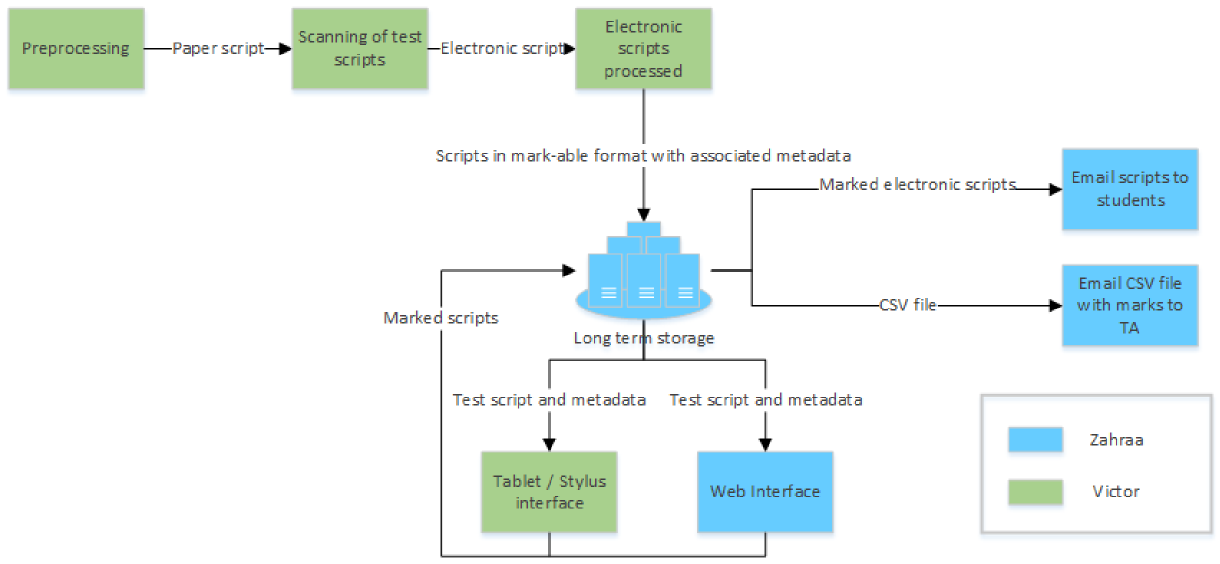

Preprocessing
Paper script
Scanning of test scripts
Electronic script
Electronic scripts processed
Scripts in mark-able format with associated metadata
Marked electronic scripts
Email scripts to students
CSV file
Email CSV file with marks to TA
Marked scripts
Long term storage
Test script and metadata
Test script and metadata
Tablet / Stylus interface
Web Interface
Zahraa
Victor

Figure 5: Overall process flow of the ScriptView system

The first step in the process is preprocessing. The preprocessing step ensures that all constraints on the physical test paper (see section 3.6.1) are adhered to before the scripts are scanned. These checks are performed by the user, but it is not time consuming since the appropriate templates for the test papers are provided and if these are used, this step can be omitted by the user.

Once scripts are collected and scanned, they are automatically emailed to a monitored inbox. As scripts arrive at the inbox, they undergo OCR in order to determine the name of the course and the test so that they can be stored on the server. This is done to remove the need for users to manually enter this information for each test which decreases the amount of work required from the user. It also reduces the probability of naming errors since all text output from the OCR methods is checked against a database of test and course information. Once the OCR is complete the script is uploaded as a set of images, each representing a single page from the script.

The act of marking a test script is largely unchanged in terms of where it fits within the process flow and how it is performed. This is done intentionally in order to decrease the amount of changes that would have to be made to the existing test management procedures if an institution wishes to adopt the use of ScriptView.

Test marks are automatically gathered, stored and analyzed by ScriptView. These marks can be accessed by administrative users and sent to students along with their test scripts. Marks can also be exported in Comma-Separated Values (CSV) format for uploading to the institutions learning management system.

3.5. Initial Software Design

Since an agile methodology with a focus on user feedback was followed during the development of ScriptView. The final product is, in some respects, vastly different from the original design. Therefore this section only documents the initial design of the software before any implementation or user feedback sessions were undertaken.

3.5.1. Test Script Processing

Test script processing involves all tasks that manipulate the students test scripts. This involves scanning the scripts; retrieving information for storage; converting them to a suitable format for the marking interfaces and converting them into a suitable format for emailing to the students.

Two approaches were considered for scanning the test scripts. The first was the creation of a dedicated desktop application which would allow the user to select a set of tests from the local storage on the machine and upload these to the server. This would allow the operation to be strictly access controlled and thus easily traceable to a single person. The disadvantages of this approach however are that the scripts would first need to be scanned and saved to the local machine, which is not only time consuming but creates a single point of failure. The uploading would also be delayed if the user forgets to start the processing on the local machine and becomes preoccupied with other administrative tasks.

The second approach was to have the scripts automatically emailed to a monitored inbox as they were scanned. This would allow for the work to be distributed among multiple individuals by allowing different people to do the scanning while still keeping the processing centralized. The process would still be traceable as each document has a timestamp that can be checked against who was using the scanner at that particular time. The machine doing the processing, would still be a single point of failure but the scanned scripts would not be lost as they would still be available in the email inbox and would be processed as soon as the applications resumes. This approach was chosen as the final design for the scanning of test scripts.

The file structure on the server requires that the course and test name of a scanned script be known. Since scripts are automatically emailed from the scanner, this information cannot be retrieved from the user at the point of scanning. It was decided that the best solution to extracting this information, would be to perform OCR on the scanned test pages.

Once the OCR has been completed the test needs to be uploaded to the server. It is at this point that it has to be converted to a format which is suitable for the marking interface. It was decided that the tests would be uploaded to the server as a set of images in Portable Network Graphics (PNG) format. This would allow for easy access to a single page within the document and was in line with the capabilities of the target device. The decision to do the conversion at this point was influenced by the limited computational power available on the target device for the mobile marking application (See section 3.6.3) as well as the goal of improving the speed at which the mobile application can provide access to the test scripts.

Before tests can be emailed to the students, the sets of images need to be converted to individual Portable Document Format (PDF) documents. This was designed as a standalone application which would be available to the server.

The diagram in Figure 6 shows the flow of a single test script through the system. This diagram was used to ensure that the script was always in the expected format at any given stage.

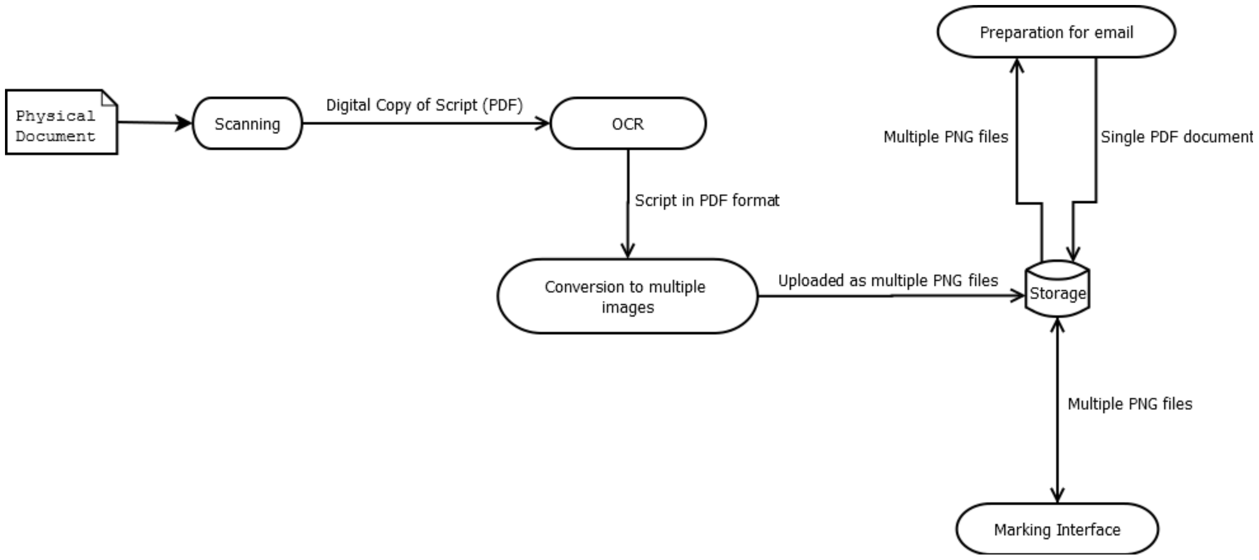


Figure 6: The flow of a script through the system

The diagram in Figure 6 was used to create the analysis class diagram in Figure 7. The aim was to design classes that would be as decoupled and cohesive as possible so that if changes needed to be made to any of the existing stages in Figure 6 or a new one added, then it would have a low impact on the number of classes that would be affected.

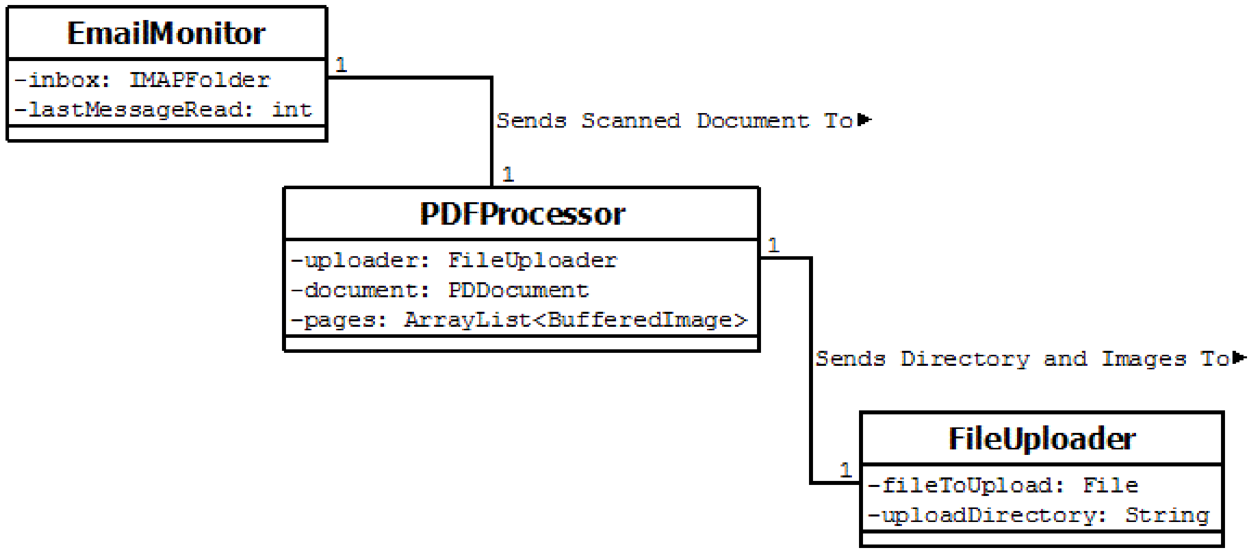


Figure 7: Analysis class diagram for the script processing component

The EmailMonitor class will handle all tasks which involve the email inbox. This includes downloading new emails as they arrive as well as downloading the PDF attachment to be sent to the PDFProcessor. The PDFProcessor will split the PDF into a series of images as well as perform the OCR on the document to determine the course and test name which will be used in the upload directory. The FileUploader will construct the upload directory, which is the test script's destination on the server, and upload the images.

3.5.2. Memorandum Processing

In order to create the mobile marking application, the questions and answers of the test is required. To avoid having the user enter this data, the memorandum will be uploaded to the server via a web interface. It will be processed by a standalone application which will need to extract the questions, answers and answer regions. The answer regions are the start and end y-coordinates of the space left open for the student to answer. For more detail on why this is required see section 3.6.2.

To accomplish this the memorandum will be split into multiple pages, similar to what is done to the scanned script, and OCR will be performed on each page. String comparison and text position will be used to categorize extracted text into question and answer categories. The position of the answer text will then be used to extract the answer regions.

3.5.3. Mobile Marking Application

The mobile marking application is the core of the three ScriptView components discussed. It is the only part that is exposed to the user and thus its design needed to be focused around usability and increasing marking speed.

As mentioned in section 3.6.3 the target device for the mobile marking application runs on the Android operating system thus the application was designed with the Android design guidelines [17] in mind. The principles which had the greatest influence on the design are discussed below:

1. **Real objects are more fun than buttons and menus**

   This principle states that it is better to allow users to directly interact with objects in the application than to have them use buttons or menus. This principle directly influenced the decision to allow the user to mark the test script through the use of gestures. Using the stylus, the user is able to annotate the test script using one of the gestures shown in Figure 8 below. Each of these already has a meaning to the marker as these are the annotations they would have made had they been marking the physical test paper. From left to right the annotations indicate a single mark, half a mark and no marks.

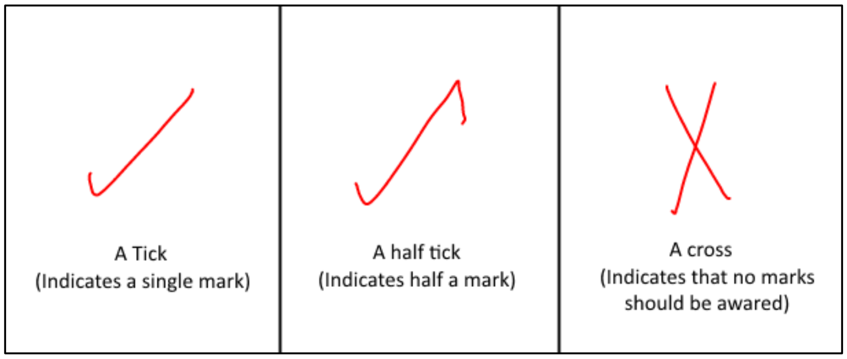

A Tick
(Indicates a single mark)
A half tick
(Indicates half a mark)
A cross
(Indicates that no marks should be awared)

Figure 8: Gestures available to the user

2. **Keep it brief**

   This principle states that instructions to the user should be kept as short as possible. Since the users of the system will have had some experience with marking test scripts. The instructions need only inform them about how to accomplish what they would usually do on paper and not the technicalities of marking. There will also not be a lengthy tutorial on how to use the application but rather tooltips for graphical elements. This guideline also influences the design of error message as the aim is to keep them brief and user friendly.

3. **Only show what I need when I need it**

   Tasks which the user cannot perform in a given context will be hidden while those that are possible will be shown using an icon rather than being hidden in a menu. This guideline also applies to the use of tooltips as the user will only need to see this additional information when they need it.

4. **I should always know where I am**

   This says that users should be confident about where they are within the app and how to navigate to previous screen. This has inspired the use of tabs at the top of the application for each of the questions that need to be marked. This will allow the user to easily move between questions beyond just moving to the previous or next question.

5. **If it looks the same, it should act the same**

   This principle ties into the concept of recognition versus recall. Which states that it is faster to interact with an object if you recognize its purpose than if you have to think about it. To aid recognition, standard Android icons will be used for common actions such as undo. Since the user encounters these in multiple other Android applications they will easily recognize their function within the marking application. However, care must be taken in ensuring that the correct actions are tied to the correct icons as an incorrect mapping will cause frustration and may lead to users becoming reluctant to use the application.

While always keeping these concepts in mind, the design of the application started at analyzing the various tasks a user may want to perform when using the application. To facilitate this the use case diagram, as shown in Figure 9, was used.

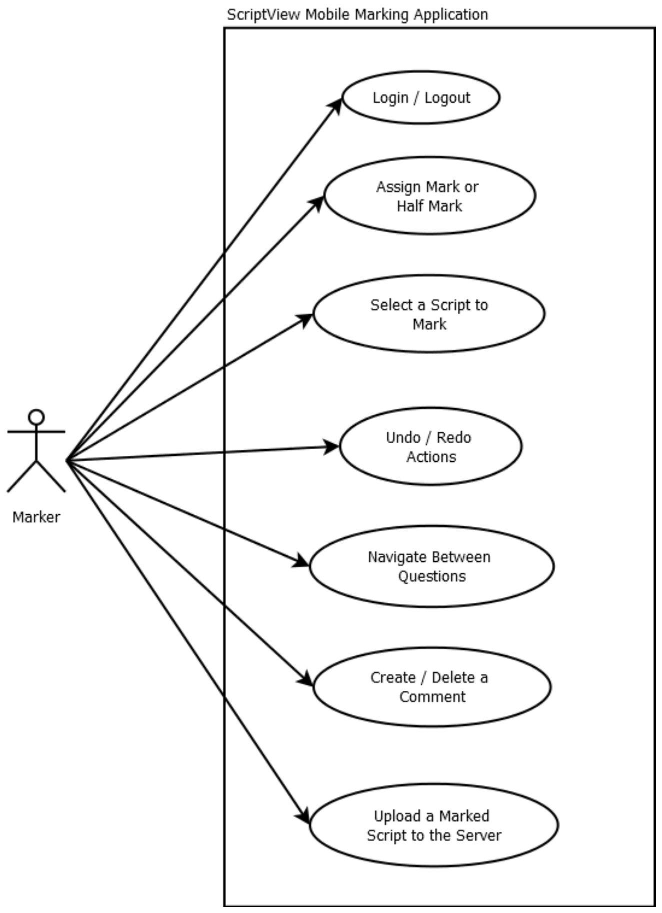

ScriptView Mobile Marking Application
Login / Logout
Assign Mark or Half Mark
Select a Script to Mark
Undo / Redo Actions
Navigate Between Questions
Create / Delete a Comment
Upload a Marked Script to the Server
Marker

Figure 9: Use case diagram of the mobile marking application

These use cases where then grouped according to how often they would be used together to ensure that all functionality on a given screen was related and would adhere to the third android design guideline stated above. Once the use cases had been grouped, the first screen that was designed was the one that users would use to mark a script. This was prioritized since it is the screen that the users use for the longest time and thus having it ready for the first user testing was beneficial.

Based on the findings from the interview with Doctor J. Tangkuampien the marking screen is designed to offer the marker a view of the question, answer and student test script (Hereafter referred to as the three-pane view). This three-pane view as seen in Figure 10 increases the markers marking speed as it always has the current question and answer in view thus decreasing the amount of work required by the marker. A mockup of the design of the marking screen can be found in Appendix B.1. As can be seen in the diagram, tabs were included in the design with the goal of allowing the user to navigate between questions and the largest area of the screen was dedicated to the student's answer since it is the focus of this screen.

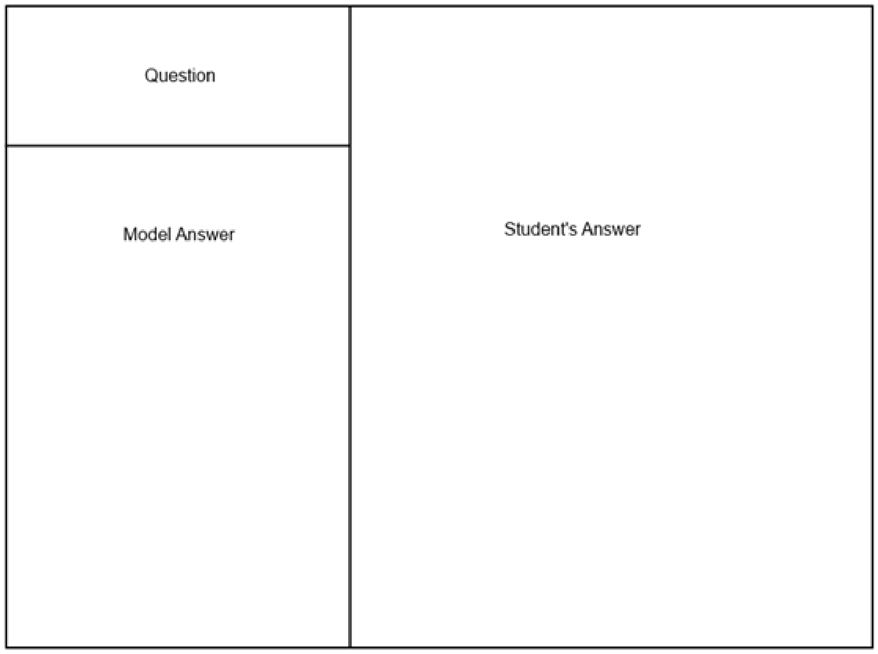


Figure 10: Three-pane view which served as a guide for interface design

The next screen which needed to be designed was the script selection screen. The purpose of this screen is to allow the user to select which test script they want to mark. The challenge presented by this screen is that it needs to show mainly navigational information such as the test and course name. Appendix B.2 shows a mockup of the design of this screen. The decision to use an Android Navigation Drawer layout was influenced by the third Android design guideline since it enables the application to hide the course information when the user is looking at tests. This was the same motivation behind using an expandable list view for the test information as scripts for another test could be hidden or shown at the users command.

The final screen that was designed was the script upload screen, a mockup of which can be found in Appendix B.3. The purpose of this screen is to allow the user to check their mark allocations, name the script according to the student number and upload the script. As can be seen in the diagram a table is used to display the mark summary with the goal of improving usability since it is a format which is familiar to markers. The screen also displays a scaled down version of the test to allow the user to verify that marks on the test paper were accurately counted.

All screens were designed to make use of the Android Action Bar, an example of which can be seen in Figure 11. This is a bar along the top of the screen which contains the application icon, action items and an action overflow menu. The decision to use the action bar with familiar icons for each action item was influenced by the fifth Android design principle.

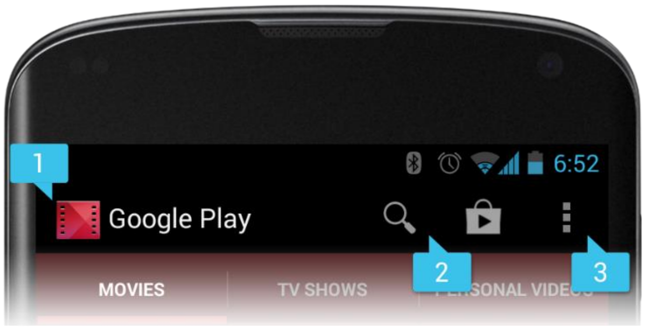


Figure 11: An example of an android action bar that includes (1) an App Icon, (2) two Action Items, and (3) an Action overflow [18]

The state diagram in Figure 12 shows which screen a user needs to be presented with at a given time. This diagram was used throughout development to ensure that data for a given screen was available at the correct times and that a user could only reach a certain screen from specific locations.

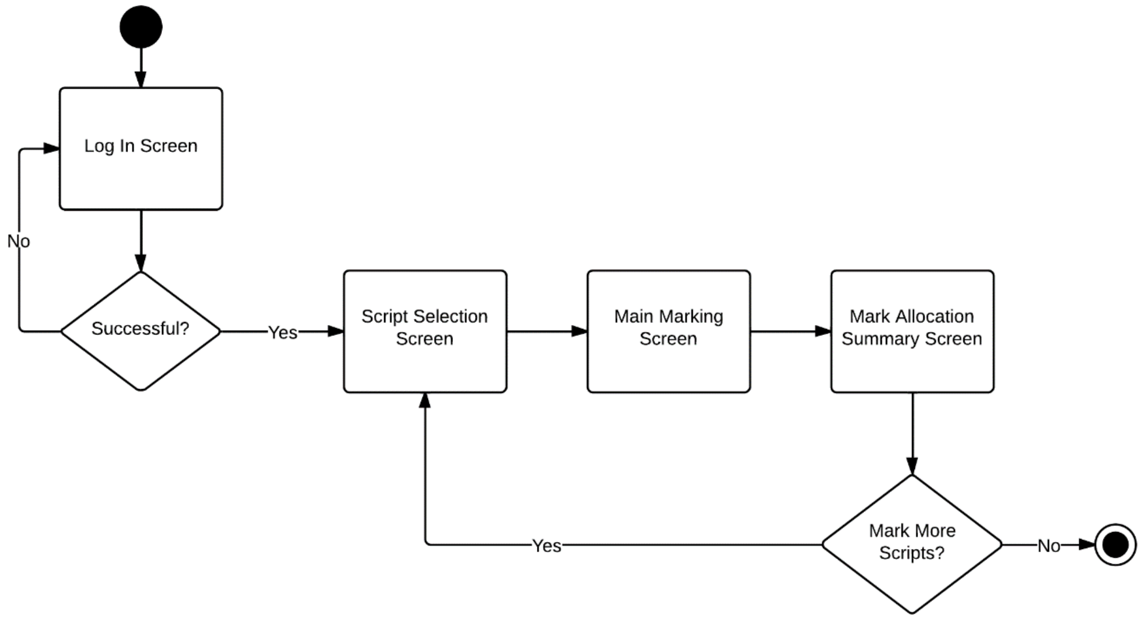


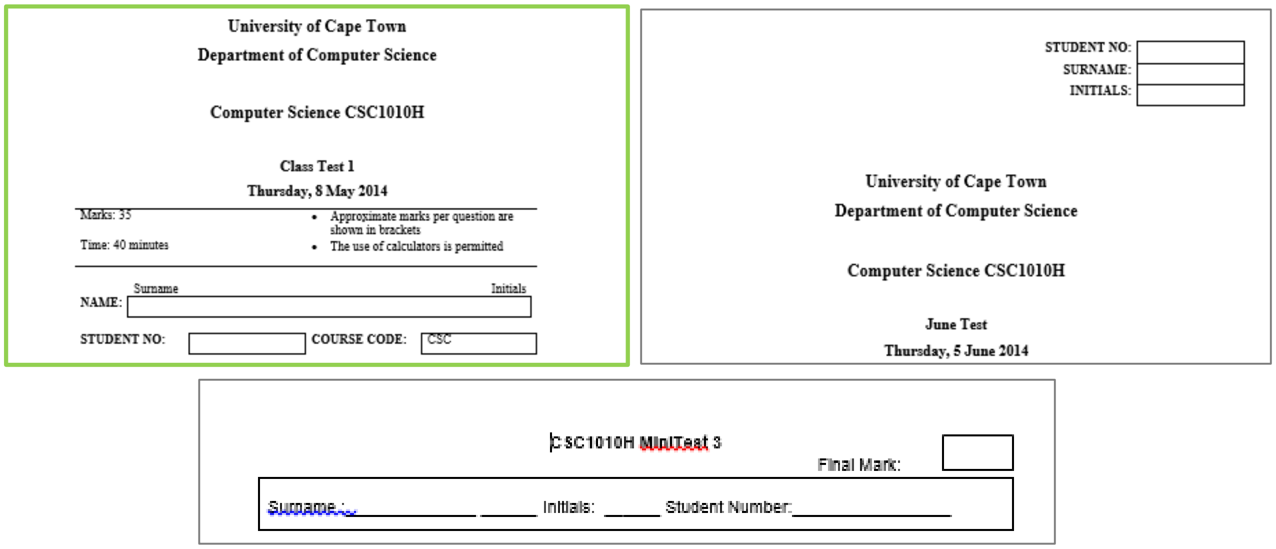
Figure 12: State diagram used to indicate screen flow of the ScriptView mobile application

3.6. Constraints

ScriptView is intended to enhance an existing procedure and as a result certain constraints needed to be established. This was to ensure that the system would be easily adopted by potential users as well as decreasing the amount of changes that would have to be made to their current workflow. This section describes these constraints and how they arose.

3.6.1. Scanned Test Script

During the design of ScriptView, sample test scripts were provided by the project supervisor. These scripts were from previous tests that were used in a single course. Figure 13 shows the variation that existed in the formatting of the first page of each of the scripts.



University of Cape Town
Department of Computer Science

Computer Science CSC1010H

Class Test 1
Thursday, 8 May 2014

Marks: 35 • Approximate marks per question are shown in brackets
Time: 40 minutes • The use of calculators is permitted

NAME: Surname Initials
STUDENT NO: COURSE CODE: CSC

STUDENT NO:
SURNAME:
INITIALS:

University of Cape Town
Department of Computer Science

Computer Science CSC1010H

June Test
Thursday, 5 June 2014

CSC1010H MiniTest 3
Final Mark:
Surname : ____________ ______ Initials: ______ Student Number: ________________

Figure 13: Variation on the cover pages of the sample test scripts

This variation meant that the scanning component of ScriptView would need to be able to handle all of them as well as any unseen cases. To accomplish this, OCR was performed on each cover page and the resulting text was searched for keywords. These keywords were related to the information which needed to be extracted, for example name, surname or test name. This approach had limited success due to the following reasons, the location of fields that needed to be completed by the student such as name and surname were accurately detected but the handwriting recognition which was then performed on these regions was unreliable and inaccurate. Text that was typed by the examiner such as the test name was accurately detected by the OCR but determining whether or not text was part of the test name required that some intelligence be incorporated into the algorithm and this was beyond the scope of the system.

In order to overcome these challenges it was decided that the formatting of the first page of the test would need to be constrained. This would allow the OCR algorithms to categories recognized text based on their position on the page and relative to each other. An example of this is that the algorithm could now determine that text after the institution name is the name of the course. Keeping the goal of reducing changes to the current procedures in mind one of the sample tests was chosen as the accepted format and only the top half of the first page was constrained. The chosen format can be seen in Figure 13 highlighted in green.

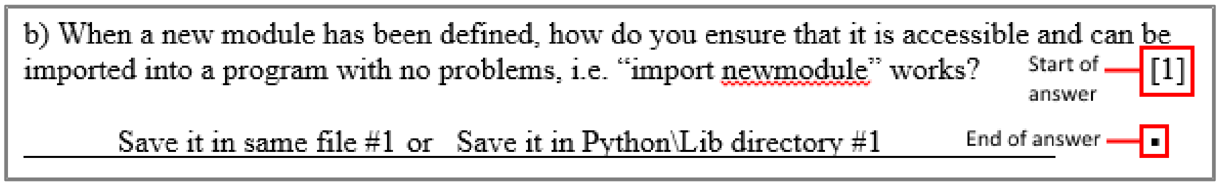
A disadvantage of this approach was that the student number was no longer detected. Since the student number is essential to the system a compromise had to be made. This compromise was that the system required the user to enter the student number before uploading the test. This was an acceptable compromise as it promotes blind marking, which is marking without information about who the script belongs to [19], and checks could be coded to ensure that the student number was accurate.

3.6.2. Memorandum

The memorandum for any test that will be marked using ScriptView, is uploaded via a web interface in PDF format. This enables the system to manipulate the way in which the questions and answers are displayed on the marking interface. In order to accomplish this, the text of both the questions and answers needs to be extracted as well as the location of the answer regions.

Extracting the text is accomplished using a similar technique as is used on the first page of the test. However, instead of using OCR the text is simply extracted from the document. This is possible since the memorandum must always be typed and is not a scanned document. Since the length of both questions and answers can vary and neither the question nor answer positions are known, categorizing the text as on the first page is not effective.

To mitigate this problem, the use of annotations on the memorandum was explored. This approach involved having the examiner place annotations at certain locations on the test paper which would indicate the start and end of an answer region. The annotation that was used to indicate the start of an answer was the mark indication which was already part of the test. To indicate the end of an answer a new annotation needed to be created. An example of these annotations can be seen in Figure 14.



b) When a new module has been defined, how do you ensure that it is accessible and can be imported into a program with no problems, i.e. "import newmodule" works? [1]

Save it in same file #1 or Save it in Python\Lib directory #1

Start of answer
End of answer

Figure 14: A question and answer with the appropriate annotations

These annotations allowed for algorithms which could accurately categorize text into either being a question or answer based on their location relative to the annotations. These annotations were also used to extract the start and end y-coordinates of the answer regions which were needed for mark allocation on the marking interface.

Although this technique was effective, through testing, it became apparent that the spacing on the memorandum and the test script that is handed to students is not always identical. This meant that the coordinates for the answer regions which were extracted would not always be accurate and would, as a result, skew any analytics performed on the test results.

To avoid this inaccuracy it was decided that the user would need to upload both the memorandum and the test script as it is handed to the students, in PDF format. By using these documents, it was possible to categorize questions and answers without the use of the annotations as answers were not present on the test handed to students so categorizing text could be achieved by comparing the extracted text from both documents. Answer region detection was then performed on the test as it is handed to students by using image processing to detect the lines left open for a student to write the answer.

The use of annotations was thus discarded but it was now necessary for the user to upload two documents instead of only the memorandum. The space left for the student's answer also needed to be indicated by lines as seen in Figure 15. Figure 15 also indicates the accepted location for the mark indication. In an effort to reduce the amount storage required, only the extracted text and answer region coordinates were stored and not the uploaded files.

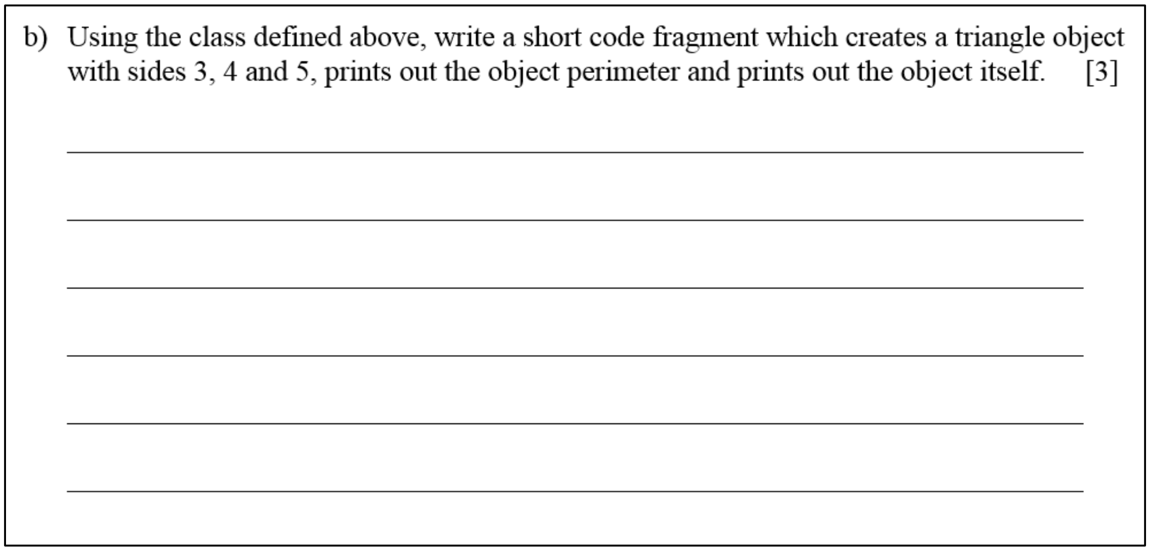

b) Using the class defined above, write a short code fragment which creates a triangle object with sides 3, 4 and 5, prints out the object perimeter and prints out the object itself. [3]

Figure 15: The accepted format for a question

3.6.3. Target Device

The final version of the mobile marking application needs to run efficiently on a Samsung Galaxy Note 10.1 and uses the Samsung S Pen stylus as input. The operating system on the device is Android version 4.1.2 (Jelly Bean).

The key constraints introduced by this device is making effective use of the 10.1" (255.8mm) screen and managing the memory when handling a large amount of images.

3.7. Project Planning

With four months to fully implement the system and the goal of involving users in the development process, a clear plan was established for all phases of the project. Since the agile methodology was used, the project was divided into four iterations. At the end of each of the first three iterations, there was an opportunity for potential users to provide feedback on the current state of the system (See section 3.3).

This feedback was on any aspect of the system irrespective of whether or not it had been fully implemented.

Tasks within each iteration were prioritized based on their importance in terms of achieving the overall goals of the software such as improved marking efficiency and reducing the probability for error. User feedback also influenced the priority of tasks as highly requested, yet in scope, features were prioritized.

3.8. Project Feasibility (First Iteration)

The first iteration of development began with an assessment of project feasibility. It was decided that to demonstrate that the system could be completed in the given time, the most complex components would need to be prototyped. These components included the image processing which needed to be performed on the scanned test scripts and the recognition of gestures made using the Samsung S Pen.

During the prototyping of the image processing algorithms the capabilities of the Open Source Computer Vision (OpenCV) library was tested. OpenCV is an image processing library with a strong focus on efficient, real-time applications and provides the user with a range of optimized image processing algorithms such as the Canny Edge Detector. Development of the prototype was done in the C++ programming language as it allowed for the fastest execution times and would thus be suitable for the task of processing tests as they were scanned.

In order to assess the feasibility of the OCR that needed to be performed on the regions detected through the image processing, Tesseract was used. Tesseract is an open-source OCR engine [20] which is compatible with a range of image formats. At this stage of the project the Linux command line tools available for Tesseract were used and no code using the library was implemented. This was sufficient to prove the feasibility of using the Tesseract engine on the scanned documents.

The gesture recognition component was coded using Java with the Samsung S Pen Software Development Kit (SDK). The SDK allows the user to make full use of the S Pen within any Android application. By using the SDK it was possible to create and save gestures to represent a tick, half tick and cross. The focus then shifted to allowing the user to perform multiple gestures on a single page and have all of them detected individually as the SDK only allowed for the detection of a single gesture on a given screen.

There was no user testing at the end of this iteration but the prototypes were presented to the project supervisors before continuing to ensure that the requirements were understood.

3.9. Core Development

Development on what would become the final product began after the prototypes from the feasibility iteration were accepted by the project supervisors. This section presents the three iterations which followed as well as how the user testing at the end of the first two impacted the design of the system. Only the results of the user testing relevant to the implementation are discussed here, a more detailed analysis of the results can be found in section 4.

3.9.1. Second Iteration

The aim of this iteration was to complete most of the core functionality that would be required to perform user testing and get constructive feedback. Thus the components that were prioritized during this iteration was the scanned test script processing, the memorandum processing and the core marking interface.

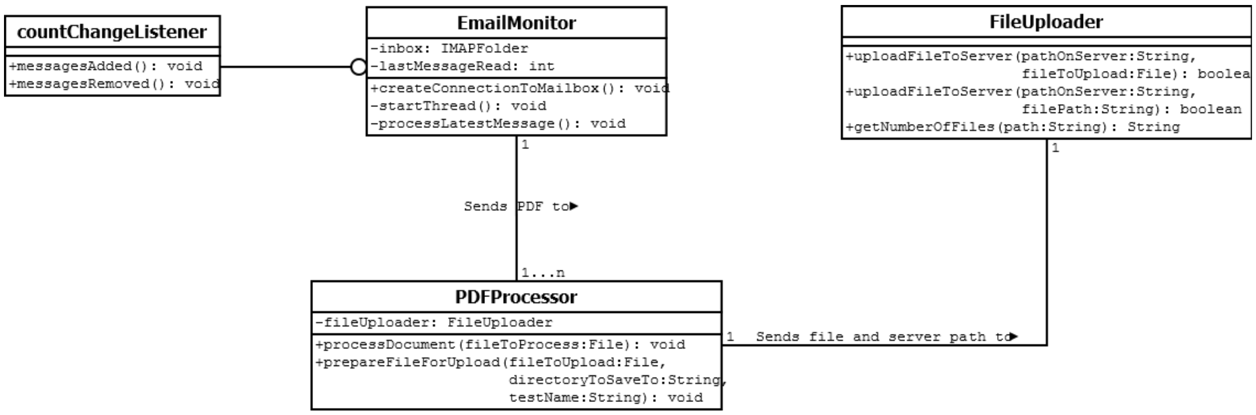
The first task that was performed during this iteration was to move all reusable code from the feasibility iteration to a repository on GitHub. GitHub is a code-hosting repository based on the Git version control system [21]. From this point onwards it was used to manage all code produced during production of the system in order to allow for effective change and bug tracking.

Work began on the script processing component by implementing techniques which allowed for the monitoring of an email inbox. To accomplish this the JavaMail Application Programming Interface (API) was used. The API provides a framework to build mail and messaging applications. It was used to establish a secure connection to a mailbox as well as download any emails that were received including their attachments. In order to prevent malicious use of this application, it can only be executed when it is on the institutional network and is connecting to a mailbox which is part of the institution. An example of this is that the UCT Computer Science issued email addresses (example@cs.uct.ac.za) can only be monitored using the application if it is running on a computer connected to the UCT network.

In order for the test script to be uploaded to the server, it had to be converted to a series of images, where each image represents a page. Since the OCR to determine the course and test name also required the first page to be converted to an image before being processed, it was decided that converting all pages to images before performing the OCR would decrease the time taken to process a single script. This was especially apparent with larger scripts as it only required a single iteration through all the pages of the script. To convert the pages to images, Apache PDFBox was used. It is a Java API which allows for the manipulation of PDF documents and was thus ideally suited for the task as the scanned scripts are emailed as PDF documents from the scanner.

The next step in the script processing was to extract the relevant information from the cover page. As mentioned before, this information is the course and test name. In order for this component to work seamlessly with the email monitoring component, it was decided that it should be implemented in Java as opposed to C++ which was used during the feasibility iteration. Since work had already been done on investigating the capabilities of the Tesseract OCR engine, Tess4j was used. Tess4j is an API which allows access to the Tesseract API to Java applications. The increase in processing time when moving from C++ to Java was investigated and was found to be negligible due to the limited amount of work required by the Tesseract API.

The final element of the script processing component was to upload all the images and the memorandum text to the server. The design class diagram in Figure 16 shows a detailed view of the class structure of the script processing component at the end of this iteration.



countChangeListener
+messagesAdded(): void
+messagesRemoved(): void
EmailMonitor
-inbox: IMAPFolder
-lastMessageRead: int
+createConnectionToMailbox(): void
-startThread(): void
-processLatestMessage(): void
FileUploader
+uploadFileToServer(pathOnServer:String, fileToUpload:File): boolea
+uploadFileToServer(pathOnServer:String, filePath:String): boolean
+getNumberOfFiles(path:String): String
1
Sends PDF to
1...n
1
PDFProcessor
-fileUploader: FileUploader
+processDocument(fileToProcess:File): void
+prepareFileForUpload(fileToUpload:File, directoryToSaveTo:String, testName:String): void
1
Sends file and server path to

Figure 16: Design class diagram of the script processing component at the end of the second iteration

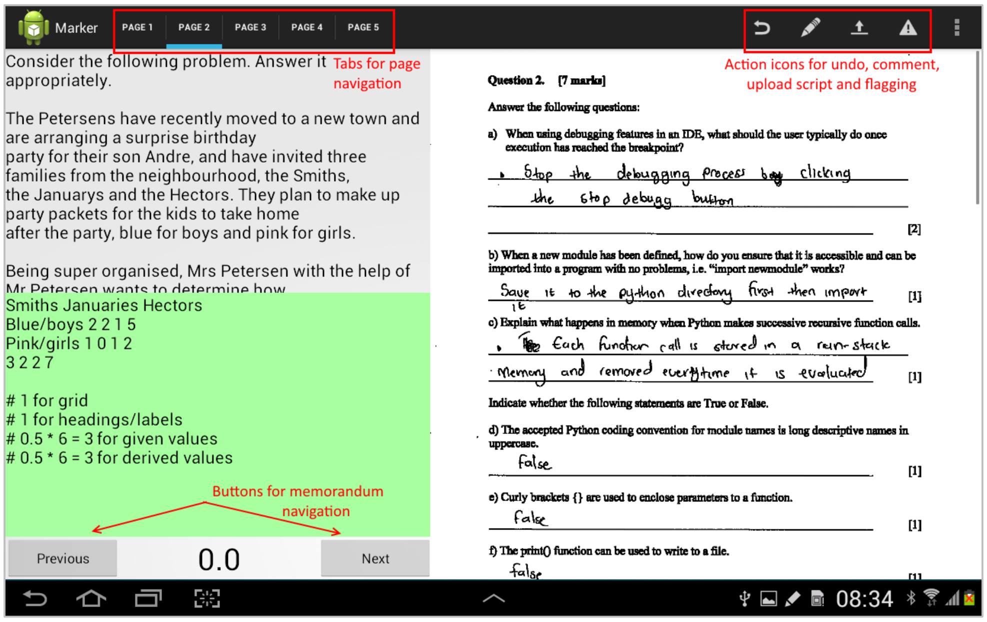
The majority of time in this iteration was focused on development of the mobile marking application. Development of the application began at the marking screen. This is the screen that users would use most often and it was expected that it would be the one to change the most based on user feedback. Completing it in time for the user testing at the end of this iteration was therefore essential.

One of the biggest challenges faced during the implementation of the mobile application was implementing the three-pane view discussed earlier. Since the design mockup in Appendix B.1 was done to scale, the challenge of effectively using the limited screen space had already been solved. The implementation challenge however, was arranging the various Android layout components, each with their own limitations on size and position, to achieve the desired effect.

The following features where also implemented on the main marking screen, the ability to undo an annotation, the ability to leave a comment and page navigation using tabs at the top of the screen. To allow the user to navigate the memorandum buttons were added to the bottom left of the screen. The appearance of the marking screen at the end of this iteration can be seen in Figure 17 on the following page.

Both the screen to select a test to mark and the screen to upload a marked test were implemented as well. One of the challenges of the test selection screen was populating the lists presented to the user with the appropriate tests available on the server. At this point in development the course and test names where retrieved from the server file structure and not the database. This was accomplished through the use of the Java Secure Channel (JSch) API. The API enabled the java application to securely connect to and browse the server using secure shell (SSH).

The JSch API was also used to enable the user to name the test on the final screen by renaming the folder on the server. It also facilitated the uploading of the marked images to the server. Once the script had been uploaded, methods were created to release the memory used by the images of the current test script.


Marker
PAGE 1
PAGE 2
PAGE 3
PAGE 4
PAGE 5
Tabs for page navigation
Action icons for undo, comment, upload script and flagging
Buttons for memorandum navigation
Previous
0.0
Next
08:34

Consider the following problem. Answer it appropriately.

The Petersens have recently moved to a new town and are arranging a surprise birthday
party for their son Andre, and have invited three families from the neighbourhood, the Smiths,
the Januarys and the Hectors. They plan to make up party packets for the kids to take home
after the party, blue for boys and pink for girls.

Being super organised, Mrs Petersen with the help of Mr Petersen wants to determine how

Smiths Januaries Hectors
Blue/boys 2 2 1 5
Pink/girls 1 0 1 2
3 2 2 7

1 for grid
1 for headings/labels
0.5 * 6 = 3 for given values
0.5 * 6 = 3 for derived values

Question 2. [7 marks]

Answer the following questions:

a) When using debugging features in an IDE, what should the user typically do once execution has reached the breakpoint?

• Stop the debugging process by clicking the stop debugg button

[2]

b) When a new module has been defined, how do you ensure that it is accessible and can be imported into a program with no problems, i.e. "import newmodule" works?

Save it to the python directory first then import it [1]

c) Explain what happens in memory when Python makes successive recursive function calls.

• ~~The~~ Each function call is stored in a run-stack memory and removed everytime it is evaluated [1]

Indicate whether the following statements are True or False.

d) The accepted Python coding convention for module names is long descriptive names in uppercase.

False [1]

e) Curly brackets {} are used to enclose parameters to a function.

False [1]

f) The print() function can be used to write to a file.

false [1]

Figure 17: The marking screen at the end of the second iteration

At the end of this iteration a user testing session was conducted. Table 1 shows the most important findings from this session.

Table 1: The findings from the first user testing session

Finding	Reasoning
The scripts are unreadable	Fifty percent of the users stated that the test script was unreadable, this was in reference to the image quality and not the student's handwriting.
The three-pane view discussed in section 3.5.3 is not as effective as initially thought	Fifty seven percent of users rated the effectiveness of the screen layout three or below which means they were undecided or thought it was ineffective. This is emphasized by the fact that eighty five percent of users felt that the interface did not improve their marking speed.
No significant changes need to be made to the stylus input other than tweaking parameters to make the gesture recognition more accurate	Seventy one percent of user said that it was easy to mark using the tablet interface which is supported by the fact that eighty five percent stated that marking with the stylus was intuitive. Many users did ask what was meant by intuitive and it was stated to them as how natural it was to use the stylus, this is, did they have to give it much thought or not.

3.9.2. Second Design Iteration

Based on the feedback received at the end of the previous iteration with regards to script readability and screen layout, it was decided that a second design iteration would be undertaken for both as the two are not mutually exclusive. This is because the size dedicated to displaying the script directly influences its readability and any processing which can be performed on the script before uploading to the server decreases the size it needs to be displayed at on the mobile application.

The second design iteration for the script processing resulted in the addition of an image dilation stage which aims to thicken the lines of both the typed and written text thus improving its readability. Since performing image processing on the scanned scripts raises ethical concerns around manipulating the student's answer, the original unprocessed version of the student's script is also uploaded to the server. The flow diagram in Figure 18 shows the updated flow of a script through the system.

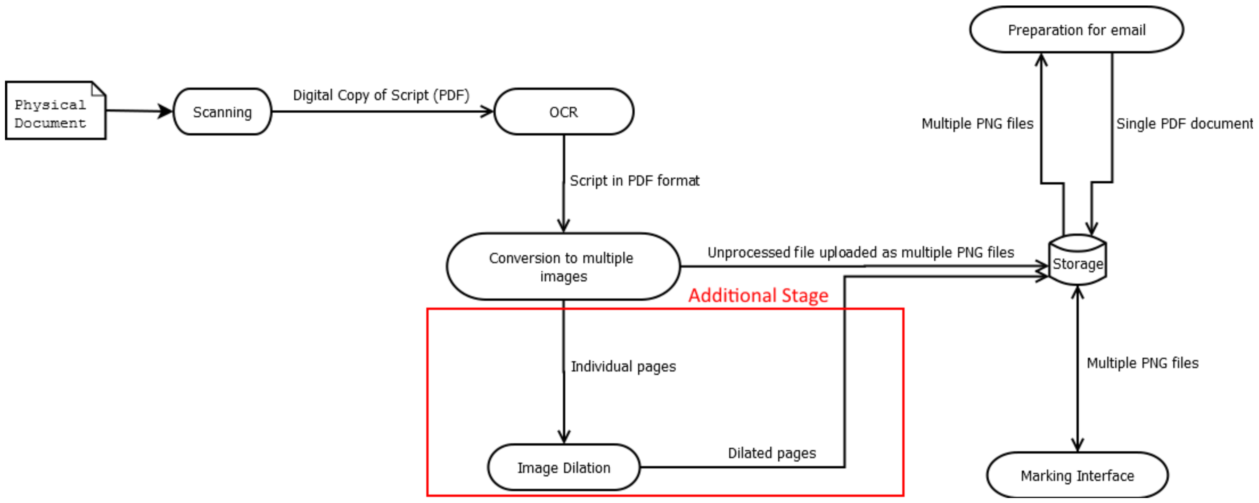


Figure 18: Updated script flow diagram after the second design iteration

The three-pane view was reevaluated and the comments received from the users was used to guide the new design. The most obvious improvement was to increase the size dedicated to displaying the test script and upon further investigation it was discovered that it was not necessary to separately display the question at all. This is because users are accustomed to reading the question on the student's script and then looking for the answer in another location. By placing the question in a different location on the screen as well, the application was actually negatively impacting the marker by showing them information they were not interested in. This realization led to the discarding of the three-pane view and a change to the template seen in Figure 19.

Users noted that they did not find having to move between memorandum answers using the next and previous buttons intuitive and would prefer if the application showed only answers which were relevant at a given time. To accommodate this, the buttons at the bottom left of the screen were discarded and the memorandum display was changed to show only the answers for a given page. The display was also designed such that users could select a given answer and the test would automatically scroll to that section of the student's test script. As can be seen in Figure 19 the memorandum answers where now designed to have clear divisions between each answer since all answers for the current page would be displayed at the same time.

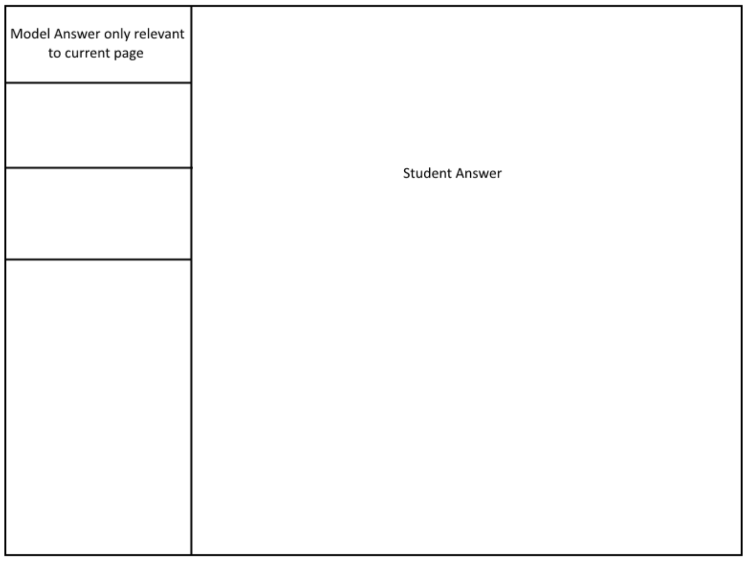
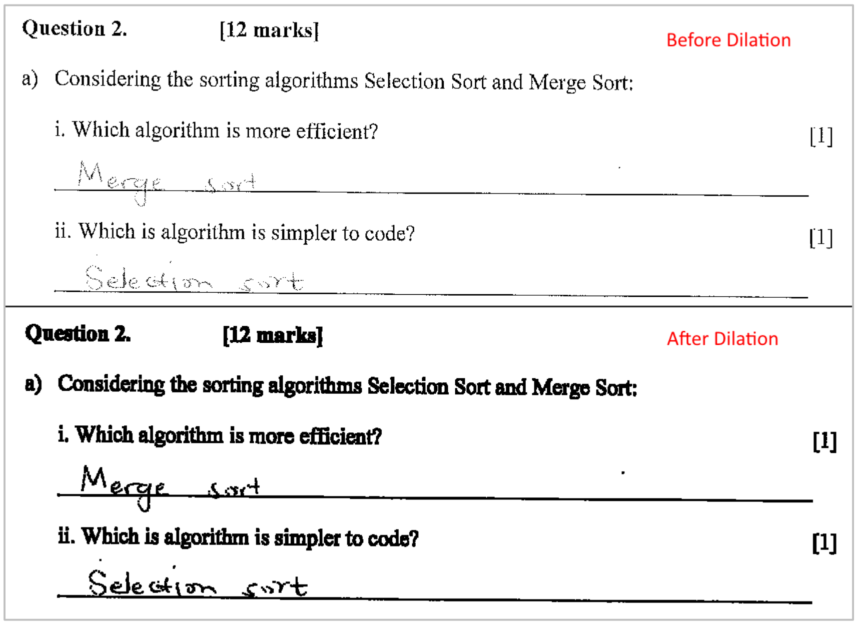


Figure 19: Marking view template after the second design iteration

3.9.3. Third Iteration

Once the second design iteration had been completed the implementation of these changes was prioritized to ensure that they would be complete in time for the final user testing session. One of these changes involved dilating the scanned image in an effort to improve readability. The effect of this on the script can be seen in Figure 20. It is clear from this diagram that the dilation not only had a positive effect on the readability of the answer but on that of the question as well.



Question 2. [12 marks]

Before Dilation

a) Considering the sorting algorithms Selection Sort and Merge Sort:

i. Which algorithm is more efficient? [1]

Merge sort

ii. Which is algorithm is simpler to code? [1]

Selection sort

Question 2. **[12 marks]**

After Dilation

a) Considering the sorting algorithms Selection Sort and Merge Sort:

i. Which algorithm is more efficient? **[1]**

Merge sort

ii. Which is algorithm is simpler to code? **[1]**

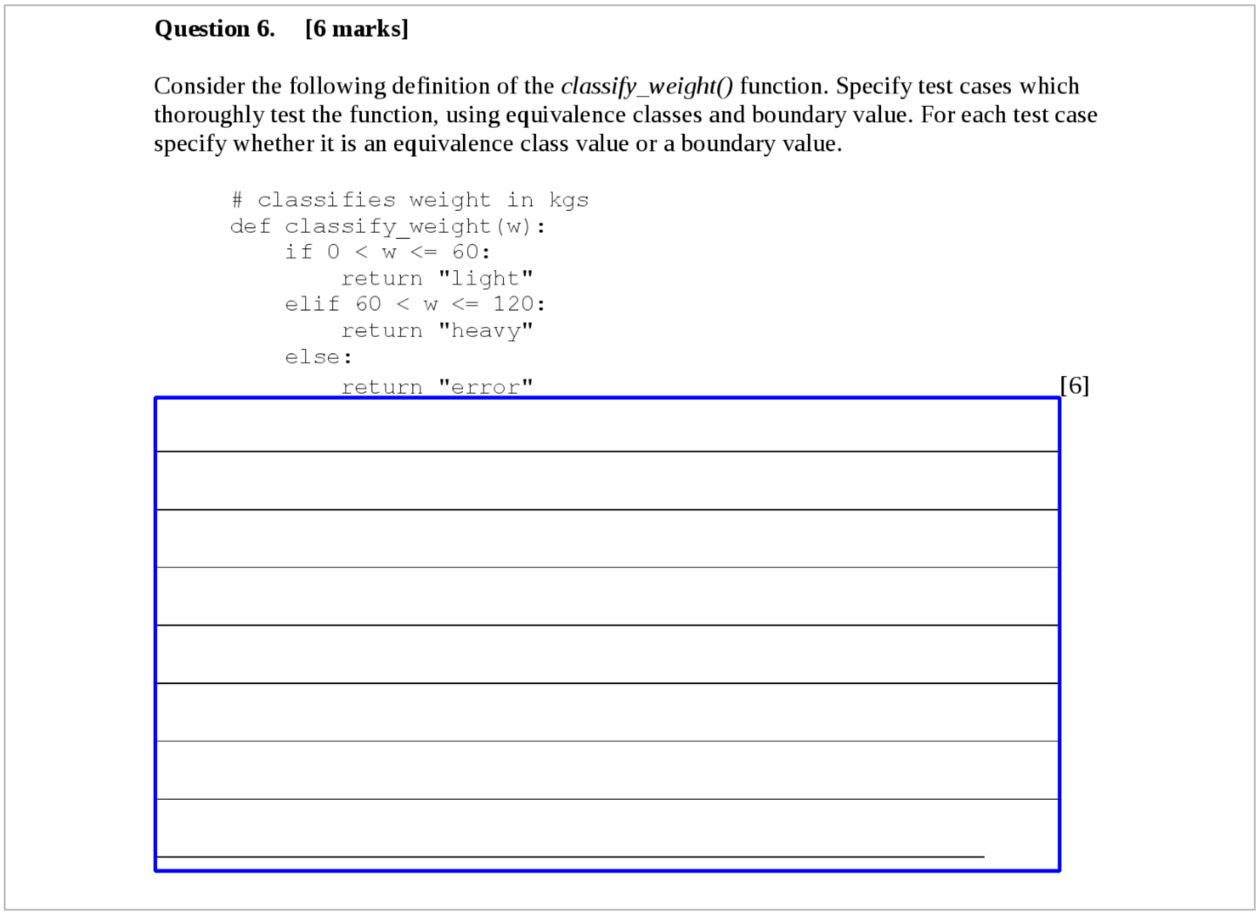
Selection sort

Figure 20: An extract from a script before (top) and after (bottom) image dilation

In addition to these changes, the mobile application was also connected to the database in order to retrieve course and student information. Connecting to the database was achieved through using PHP Hypertext Processor (PHP) code which was hosted on the server. Using PHP allowed for highly reusable code which could be accessed at various points in the application.

Connecting the application to the database also enabled the development of predictive text and rigorous error checking when the user names the test scripts. This greatly reduced the possibility of naming errors. With the application connected to the database functionality such as uploading the student's mark, flagging a script for follow up and access control was implemented.

Detecting answer regions on the memorandum was also implemented during this iteration. In order to detect the answer regions on the memorandum the OpenCV implementation of the Hough line transform was used. The Hough line transform is a transform which is used to detect straight lines. This suited the application well since answer regions are demarcated as a collection of horizontal lines on the test script. Figure 21 shows the answer region, the blue rectangle, which was detected when a script was provided as input. The detected region is larger than the collection of lines since additional space was explicitly added to the top and bottom of the region to improve accuracy when used within the marking application.



Question 6. [6 marks]

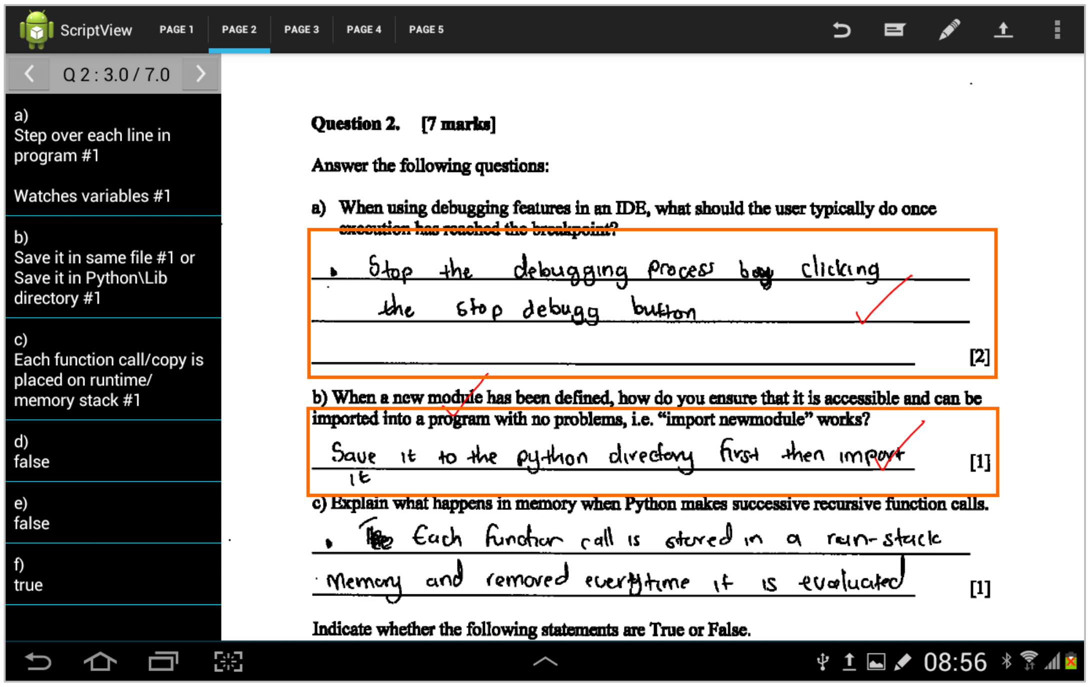
Consider the following definition of the *classify_weight()* function. Specify test cases which thoroughly test the function, using equivalence classes and boundary value. For each test case specify whether it is an equivalence class value or a boundary value.

```
# classifies weight in kgs
def classify_weight(w):
    if 0 < w <= 60:
        return "light"
    elif 60 < w <= 120:
        return "heavy"
    else:
        return "error"
```

[6]

Figure 21: The answer region (in blue) detected by using Hough line transform on a script

In order to collect information on mark allocation beneficial to the analytics which need to be performed, it was necessary to determine to which sub-question a mark was assigned. This was accomplished by taking the median y value of the annotation and determining if it falls within a certain answer region. If the annotation was not assigned to a sub-question then it was matched to the region to which it was closest. By using this method an accuracy of 95% for mark allocation was achieved. The orange rectangles in Figure 22 represent the answer regions and based on the median y value the annotation which lies between the two will be assigned to the lower one. Since the annotation was placed on the text of question b, there is a high probability that this allocation is correct.



a)
Step over each line in program #1

Watches variables #1

b)
Save it in same file #1 or Save it in Python\Lib directory #1

c)
Each function call/copy is placed on runtime/ memory stack #1

d)
false

e)
false

f)
true

Question 2. [7 marks]

Answer the following questions:

a) When using debugging features in an IDE, what should the user typically do once execution has reached the breakpoint?

• Stop the debugging process by clicking the stop debugg button

[2]

b) When a new module has been defined, how do you ensure that it is accessible and can be imported into a program with no problems, i.e. "import newmodule" works?

Save it to the python directory first then import it [1]

c) Explain what happens in memory when Python makes successive recursive function calls.

• ~~The~~ Each function call is stored in a run-stack Memory and removed everytime it is evaluated [1]

Indicate whether the following statements are True or False.

Figure 22: Mark allocation to sub-questions based on the proximity of annotations to answer region

A problem introduced by this method however was that it slowed down the gesture recognition. Even though this increase in processing time was not noticeable to the user, it did have implications for the accuracy of the final mark. This is because gesture recognition is not performed on the main application thread and multiple gestures could be recognized in parallel. This meant that one thread could read and update an old mark value which could then be overwritten by another thread which had also read the old value, essentially discarding one of the updates. To solve this problem, access to the marks was restricted to a single method which used the Java synchronized keyword. This prevented multiple threads from accessing the mark value at the same time by causing threads to block if another thread was modifying the mark value.

With changes to the mobile application complete, improvements to the script processing needed to be made which only became apparent after using it during the first user testing. These improvements included having to handle scripts on which the OCR was not successful or scripts that were incorrectly uploaded to the server. Both of these errors were mitigated by using a backlog of scanned scripts, an upload window and the database of course information. The upload window was defined as the time during which a successful upload had occurred. The state diagram in Figure 23 shows how these two components were used.

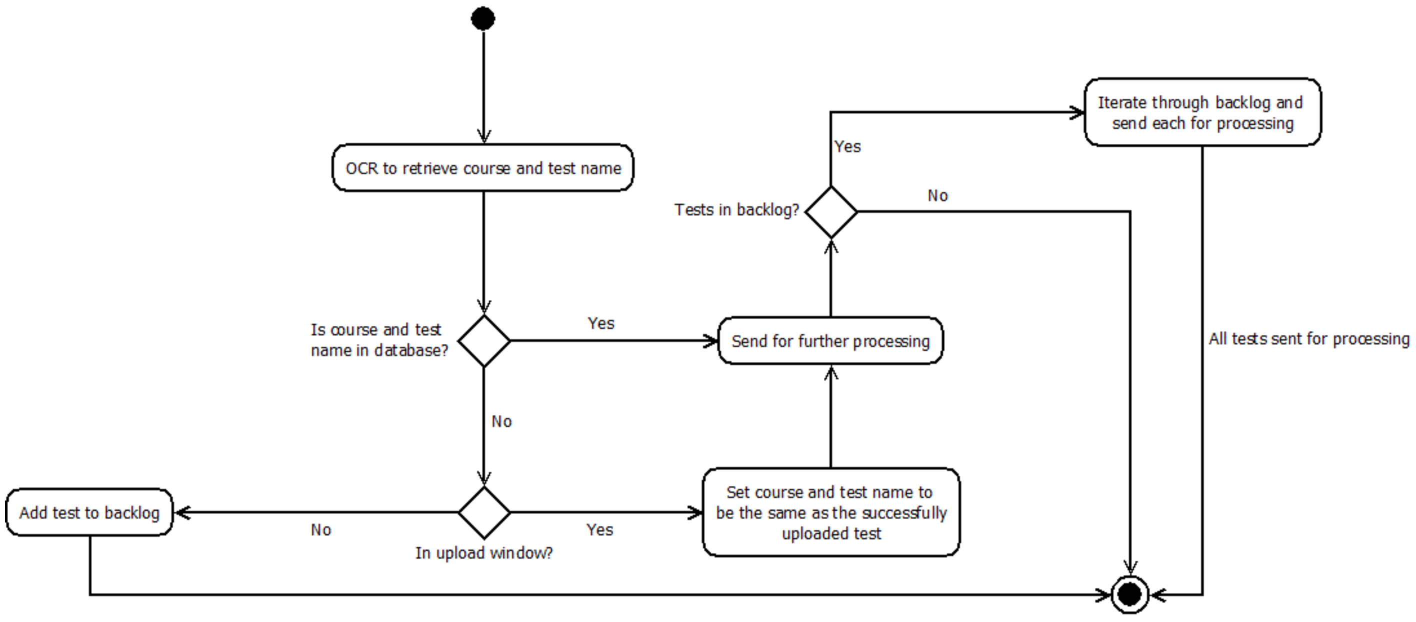

OCR to retrieve course and test name
Is course and test name in database?
Yes
No
Send for further processing
Tests in backlog?
Yes
No
Iterate through backlog and send each for processing
All tests sent for processing
Add test to backlog
No
Yes
In upload window?
Set course and test name to be the same as the successfully uploaded test

Figure 23: State diagram showing use of backlog and upload window

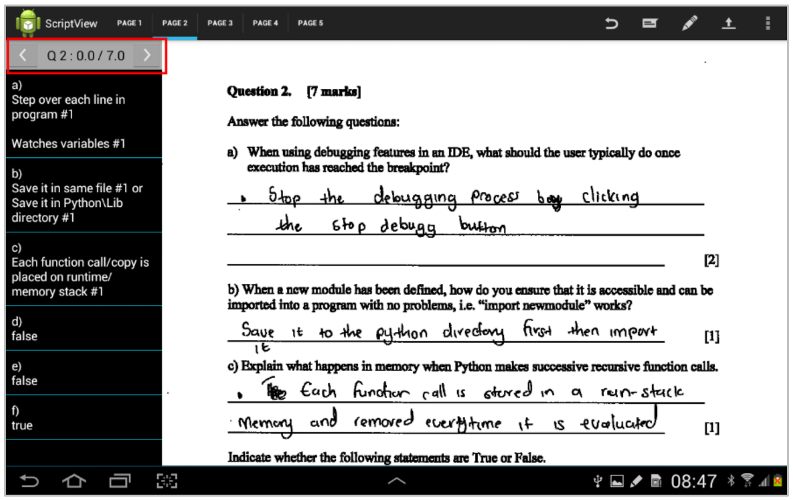
A second user testing session was conducted at the end of this iteration. Since the application was more feature complete for this session, there were more User Interface (UI) elements on the screen and as a result user feedback was focused around interaction and feedback.

Table 2: The findings from the second user testing session

Finding	Reasoning
There is a problem with detecting crosses	This is based on the fact that many users raised this concern verbally while using the system.
The display of marks per page is not intuitive	Users expressed their concerns about the application displaying the marks per page as opposed to marks per question which is what they were expecting.
Error messages need to be clearer	There were users which felt that the displaying of error messages needed to be improved as they were often too small or stayed on the screen for too short a time.

3.9.4. Fourth Iteration

Based on the feedback received during the previous iteration changes in this iteration involved changing the display of marks to conform to what users expected as well as allowing them to navigate between the marks for each question on the marking screen. The mark display is highlighted in red in Figure 24, the white arrows on either side of the display allow the user to navigate between the marks for each question without having to change the page that is being displayed. A warning will also be presented to the user should they allocate more marks than what is allowed for the question.



a)
Step over each line in program #1

Watches variables #1

b)
Save it in same file #1 or Save it in Python\Lib directory #1

c)
Each function call/copy is placed on runtime/ memory stack #1

d)
false

e)
false

f)
true

Question 2. [7 marks]

Answer the following questions:

a) When using debugging features in an IDE, what should the user typically do once execution has reached the breakpoint?

• Stop the debugging process ~~bay~~ clicking the stop debugg button

[2]

b) When a new module has been defined, how do you ensure that it is accessible and can be imported into a program with no problems, i.e. "import newmodule" works?

Save it to the python directory first then import it [1]

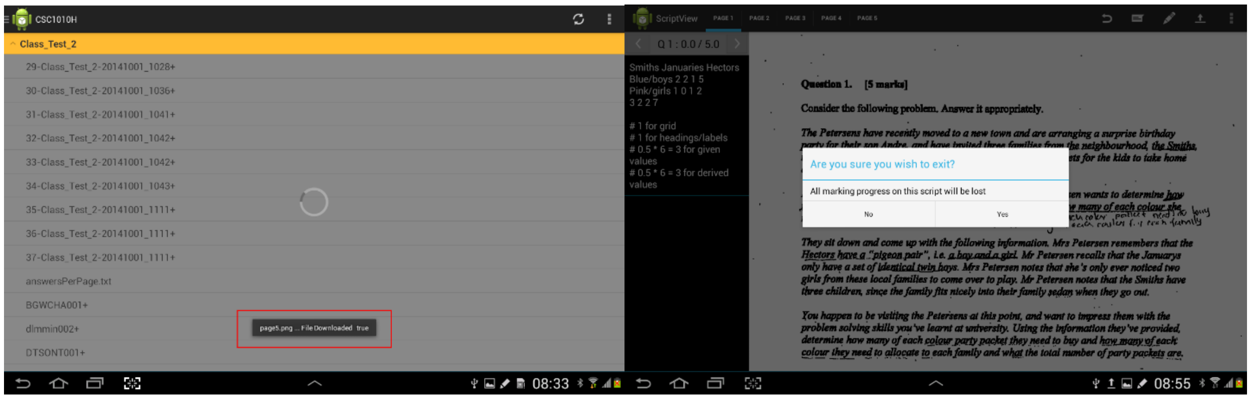
c) Explain what happens in memory when Python makes successive recursive function calls.

• ~~The~~ Each function call is stored in a run-stack memory and removed everytime it is evaluated [1]

Indicate whether the following statements are True or False.

Figure 24: Updated mark display

Changes were also made to the display of errors as they are now displayed using Android AlertDialogs as opposed to Android Toasts which were used before. The advantage of using AlertDialogs is that they need to be dismissed by the user thus ensuring that they have been seen whereas the Toasts would appear at the bottom of the screen for a short duration and then disappear. An example of each of these can be seen in Figure 25.



Class_Test_2
29-Class_Test_2-20141001_1028+
30-Class_Test_2-20141001_1036+
31-Class_Test_2-20141001_1041+
32-Class_Test_2-20141001_1042+
33-Class_Test_2-20141001_1042+
34-Class_Test_2-20141001_1043+
35-Class_Test_2-20141001_1111+
36-Class_Test_2-20141001_1111+
37-Class_Test_2-20141001_1111+
answersPerPage.txt
BGWCHA001+
dlmmin002+
DTSONT001+

page5.png ... File Downloaded true

Smiths Januaries Hectors
Blue/boys 2 2 1 5
Pink/girls 1 0 1 2
3 2 2 7

1 for grid
1 for headings/labels
0.5 * 6 = 3 for given values
0.5 * 6 = 3 for derived values

Question 1. [5 marks]

Consider the following problem. Answer it appropriately.

The Petersens have recently moved to a new town and are arranging a surprise birthday party for their son Andre, and have invited three families from the neighbourhood, the Smiths, ets for the kids to take home

en wants to determine how w many of each colour she

Are you sure you wish to exit?

All marking progress on this script will be lost

They sit down and come up with the following information. Mrs Petersen remembers that the Hectors have a "pigeon pair", i.e. a boy and a girl. Mr Petersen recalls that the Januarys only have a set of identical twin boys. Mrs Petersen notes that she's only ever noticed two girls from these local families to come over to play. Mr Petersen notes that the Smiths have three children, since the family fits nicely into their family sedan when they go out.

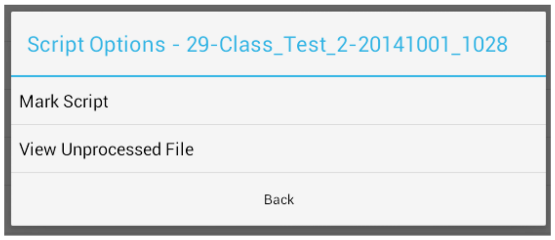
You happen to be visiting the Petersens at this point, and want to impress them with the problem solving skills you've learnt at university. Using the information they've provided, determine how many of each colour party packet they need to buy and how many of each colour they need to allocate to each family and what the total number of party packets are.

Figure 25: An example of a Toast (left) and an AlertDialog (right)

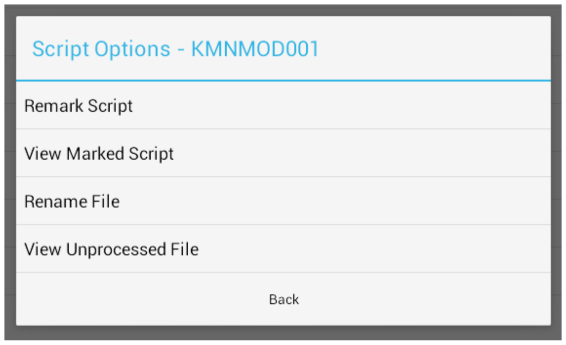
In order to address the fact that crosses were being counted as ticks, additional logic had to be added before the standard gesture detection. The first step in this process was to determine if the current gesture being processed was intersecting the previous one. To ensure that the application was still responsive, the intersection test was done by creating a bounding box around each gesture and checking for the intersection of the rectangles. The more computationally expensive step of determining whether or not two gestures have points in common was only performed if the bounding rectangles intersected. In order to determine if this intersection was in fact a cross, the two gestures only had to have a single point in common. By using this method a high degree of accuracy was achieved. Although the method does not yield an accuracy of 100%, it is acceptable due to the variation which exists in handwriting and the difficulty involved in trying to accommodate all variations.

The stability of the application was also improved by ensuring that all data is handled correctly and remains consistent if the internet connection is lost. Methods were also created to handle the case where a marker may provide a name for the script which conflicts with another already on the server. In this case the script is uploaded using a preserved prefix which indicates an error and the script is flagged so that an administrative user can resolve the problem.

Limited administrative features were also added to the tablet interface. The features which were implemented were chosen based on whether or not they were directly related to script manipulation and how often a user may want to use it on the tablet. Since features are related directly to scripts and not manipulation of the marks, the options are presented in a pop-up dialog when the user selects a script. To improve usability the options are context sensitive meaning that only options relevant to the selected script are displayed, an example of this can be seen in Figure 26.



Shown for an unmarked script



Shown for a marked script

Figure 26: A comparison between the options shown for a marked (left) and unmarked (right) script

Changes to script processing were also made which allowed for the scanning of multiple scripts at the same time. The script processing automatically detects when multiple tests are scanned by using the results of the OCR and the database of course and test information. This was an effort to further decrease the amount of work required by the person responsible for scanning the documents. The final stage of script processing which involves converting the marked image files of a test script into a single PDF file, which can be emailed to the student, was also implemented.

Development of the ScriptView system was completed at the end of this iteration.

4. Results

The results which had the greatest impact on the development of the application are briefly mentioned in Section 3.9 this was done in order to provide some context as to why certain design decisions were made. The purpose of this section is to present the quantitative data collected from the user testing sessions as well as summarize some of the most common written responses. The questionnaire used can be found in Appendix A.

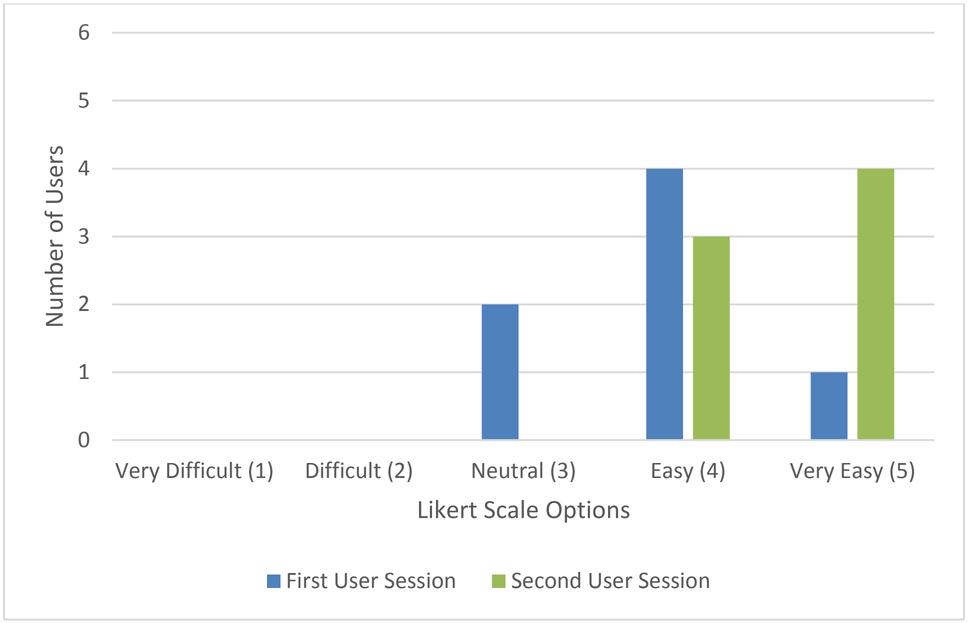


Figure 27: Users' responses to the difficulty of using the interface

Figure 27 shows how users felt about the difficulty of using the interface. From the graph it is clear that users found the interface easier to use during the second user session than the first.

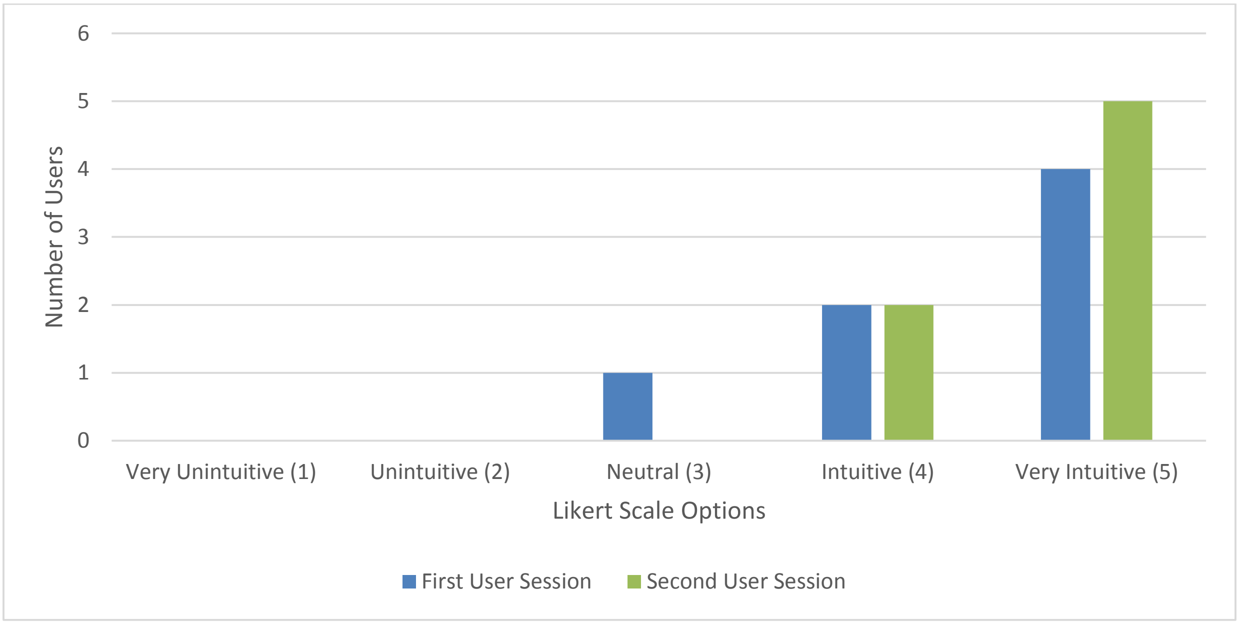


Figure 28: Users' responses to the intuitiveness of using the stylus

From Figure 28, which shows how intuitive users found using the stylus, there is also an increase between the two user sessions. This increase however, is not as significant as users already found the stylus intuitive during the first user session.

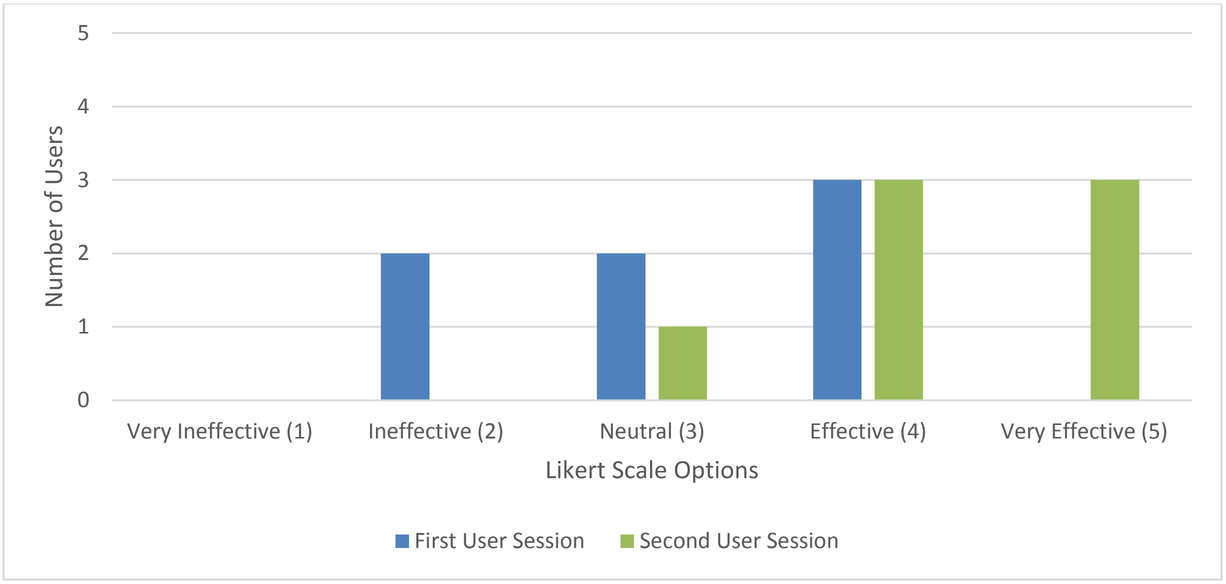


Figure 29: Users' responses to the effectiveness of the screen layout

As can be seen in Figure 29 the results for screen layout were fairly divided during the first user session but became more positive during the second with no users rating it as ineffective.

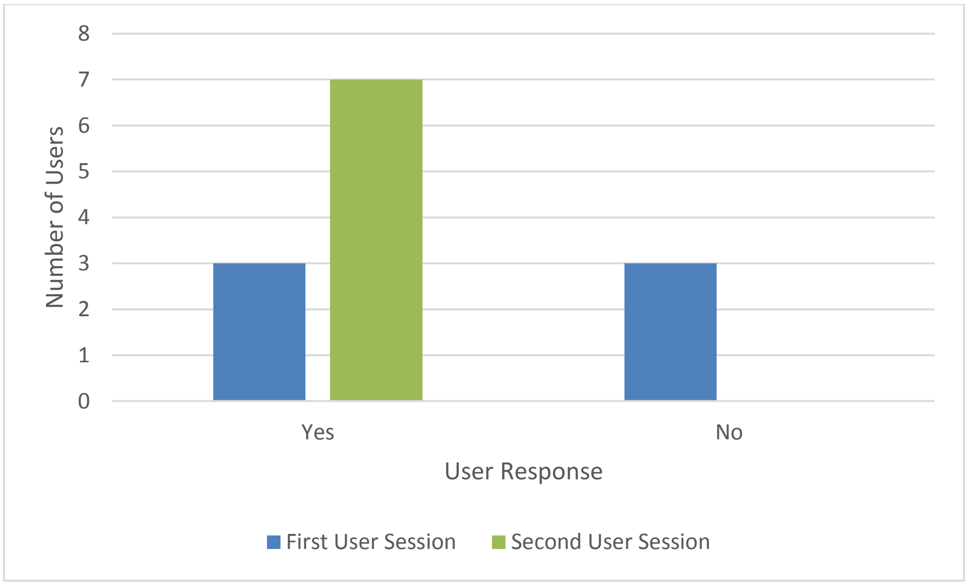


Figure 30: Users' responses to whether or not the student's script was readable

Figure 30 shows the users' responses to script readability. As can be seen from the graph, the results were evenly divided during the first user session but it is worth noting that one user did not provide an answer to this question and was thus not counted. Upon further discussion with the users it was discovered that most of them found the scripts unreadable despite indicating ‘yes’ on the questionnaire. There is drastic increase in positive responses between the first and second user session, as all users rated the script as readable during the second session. This was consistent with their responses during discussions held after completing the questionnaires.

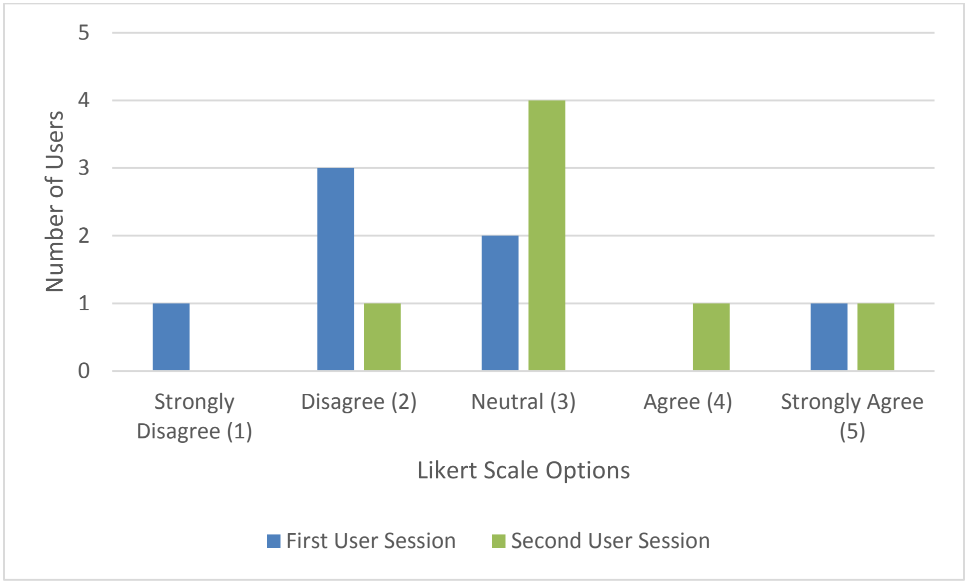
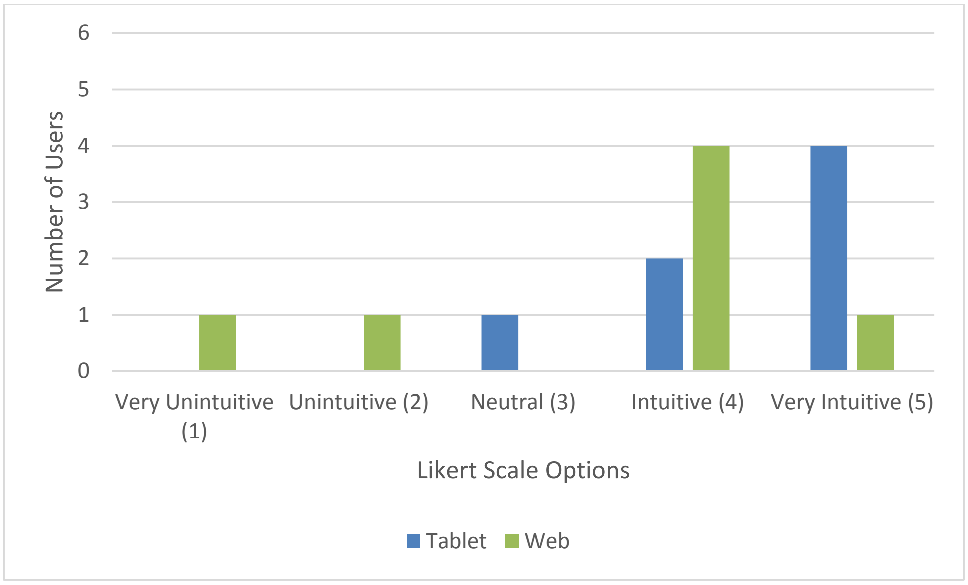


Figure 31: Users' responses to whether or not they felt that the system improved their marking speed

From Figure 31 it appears that users did not feel like the system improved their marking speed during the first user testing. Although responses seem more positive during the second user testing, they are still reasonably bad when compared to the increases seen in other aspects of the system.

From Figure 32 it is clear that user response shifted towards being more positive during the second user testing. The tablet interface however, received more positive responses during both sessions whereas the results for the web interface were more distributed during the first user session.

As for the written responses, during the first user session, users expressed concerns about gesture recognition and screen layout. While during the second session, they expressed concerns about the clarity of instructions within the application.



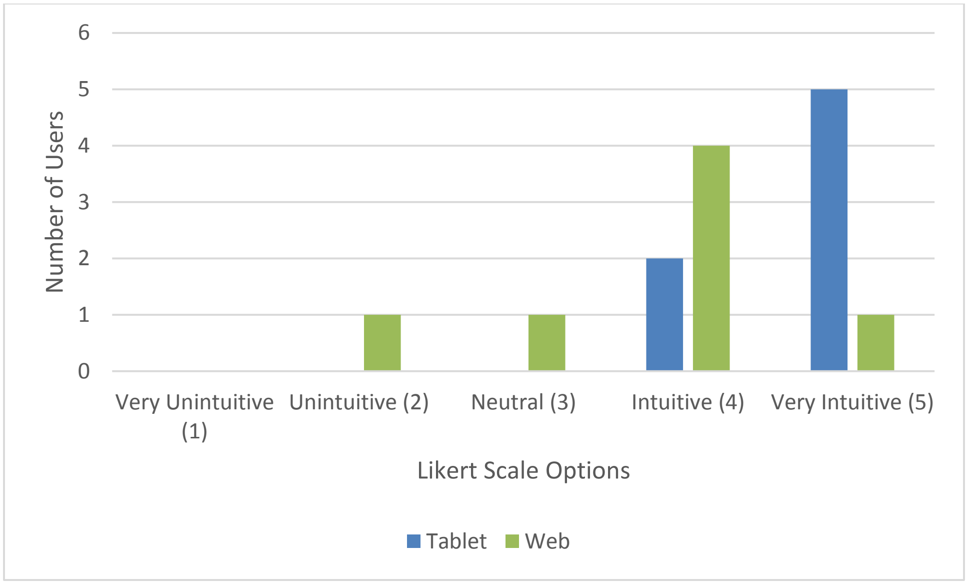


Figure 32: A comparison between users' responses to interface intuitiveness for the tablet and web interface for the first (top) and second (bottom) user session

4.1. Discussion

Based on the increase in positive responses for script readability from the first to the second iteration it is clear that the image dilation that was added to the script processing during the third iteration, was successful. However, the reasonably low results for whether or not the user felt that the system improved their marking speed raises the concern that the system would not be used by tutors as they would not see the benefit in it. These low scores could however, be due to the lack of exposure users have had with the system as the responses from the second user testing are generally more positive. This is reflected in the fact that more users found marking with the stylus intuitive, gave it a rating of 4 or above, during the second user test than during the first.

The increase in the rating of screen layout shows that the redesign of the interface from the three-pane view template was effective. This value however, could also have been influenced by the user's previous exposure to the Android operating system and thus their experience at using it. This is because standard Android icons and interaction techniques were used.

When compared to the results of the web interface, the tablet and stylus interface was generally rated more positively with respect to intuitiveness. Although the tablet interface still performed better during the second user testing, the results for the web interface also improved. The low scores for the web interface during the first user testing could thus be attributed to the fact that it was very foreign to the users. This is because it is very different from how they traditionally mark as opposed to the tablet which more closely mimics what they do when marking the paper script. This is also why the results were slightly more positive during the second user testing as users were now more accustomed to the web interface.

5. Ethical and Legal Concerns

The data which ScriptView manipulates is considered sensitive information as it affects the academic standing of the student to whom the data belongs. This had to be considered during both system development and system testing.

5.1 System Implementation

The ethical concerns surrounding the management of test scripts affects the entire lifespan of the system and thus features were put in place to maintain the integrity of the data. One of the first concerns was the secure storage of the scanned test scripts. Multiple options were considered to accomplish this but the most secure solution which could be found was the server hosted by the UCT Department of Computer Science. By using this server access to the scripts could be strictly access controlled not only by the application but also the infrastructure already in place which only allows authenticated UCT staff to access the server.

The image processing on the test script which aims to improve readability meant that students could claim that there script had been tampered with and thus their results are not valid. To mitigate this, the unprocessed test script is also stored on the server alongside the processed one. This provides administrative users with easy access to the script should the integrity of the results or test script be questioned.

To promote blind marking, the student number on the test script is hidden from the marker until they have completed marking the script. One of the ethical concerns surrounding the mobile application which has unfortunately not been fully addressed in the implementation, is preventing users from making copies of the test script. While all files are deleted from the devices local storage as soon as they have been loaded into memory, the user is still able to take screenshots. This is because it is a feature which is built into the operating system and is available across all applications.

5.2 System Testing

In order to accurately develop and test ScriptView, the use of real tests was essential. In order to do this twenty seven students from the Computer Science CSC1010H course provided consent for the use of their test scripts. These test scripts were from the second and third class tests of the course. Consent also had to be acquired from the tutors directly involved with the user testing in order for the data to be presented in this report.

5.3 Legal Considerations

In order to conduct this research ethical clearance needed to be acquired from the University of Cape Town and the Faculty of Science Research Ethics Committee within the university. Clearance also needed to be acquired to access student data and perform user testing with staff.

All the relevant legal documentation can be found in Appendix C.

6. Conclusion

The aim of the project was to enhance the current test management procedures by utilizing technology at specific stages in the process. The components described in this report are related specifically to processing the paper test script and memorandum as well as creating a mobile tablet based application for marking the scripts.

By not focusing on a completely automated solution the system allows users the freedom they have during the current procedures such as setting any type of questions and having these accurately assessed. The system also creates new opportunities such as allowing markers the freedom to mark wherever they want as opposed to having to meet at a single location. The examiners can allow this since access to the application is access controlled and thus allows them to more accurately trace who marked a test script and when.

In terms of whether or not ScriptView effectively increases marking speed, no conclusions can be drawn from the user testing as users will become more efficient at using the system over time. However, by following an agile methodology with a focus on user input, the final system has an increased potential of being accepted by users. This is further emphasized by the positive responses received about the intuitiveness of marking using a tablet and stylus interface. Although the responses for both the web and tablet interfaces were fairly similar, it would appear that users prefer the tablet interface as it more closely resembles how they usually mark.

Based on the results it is thus clear that a tablet and stylus interface is considered intuitive by markers and is preferred over the web based interface.

7. Future Work

This section describes the components of ScriptView which can be improved. These improvements were not included in the final system due to time and resource constraints.

- Compatibility with any cover page layout

  This would include creating a more intelligent script processing algorithm which could locate and extract the course and test name irrespective of the cover page layout. This could be extended to do accurate handwriting recognition on the student number and thus remove the need for it to be manually entered by the marker.

- Accurately extract diagram from the memorandum

  The current version of ScriptView is limited in that it only accurately extracts text answers from the memorandum. This improvement would require extracting regions of the memorandum as images as well as changing the marking interface to display these while still maintaining script readability.

- Automated detection of comments

  The current system requires users to specify when they would like to write a comment on the test script by selecting an option within the application. This could be simplified by adding gesture detection for characters so that, comments are automatically distinguished from marking gesture such as the tick, half tick and cross.

- Allow for simultaneous marking of a single script

  Managing test scripts which need to be marked by different individuals is still a bottleneck in the process, even though this is simplified by ScriptView. A feature could be created which would allow users to mark different parts of the test script at the same time.

- A chat functionality within the mobile marking application

  Since the application allows markers the freedom to mark wherever they want, a chat functionality would allow them to discuss and resolve problems such as ambiguity without having to flag a script.

- Notify an administrator of scanning errors

  The current implementation of the system will not notify administrators if there are scripts left on the backlog, but will provide them with access to it. By using a similar method to the script flagging, an administrative user could be notified about any errors which occurred during the scanning process on the dashboard of the web interface.

- Allow for more administrative control from the tablet interface

  Many of the administrative tasks such as emailing marks to students or viewing the output of the analytics performed on the marks can currently only be done from the web interface. It would be beneficial to the system if these features are added to the tablet interface.

- Allow for the uploading of scripts in the background

  One of the aims of the application is to improve marking speed and by adding this feature, a user would be able to continue marking another script while the previous one is uploaded to the server.

8. References

1. Poor, D. D. (1995). Image Capture and Storage Techniques in Association with Optical Mark Reading

2. Christie, J. R. (1999). Automated essay marking-for both style and content. *Proceedings of the Third Annual Computer Assisted Assessment Conference, Loughborough University, Loughborough, UK*

3. Pulman, S. G., & Sukkarieh, J. Z. (2005). Automatic short answer marking. *Proceedings of the Second Workshop on Building Educational Applications using NLP,* 9-16.

4. Thomas, P. (2003). The evaluation of electronic marking of examinations. *ACM SIGCSE Bulletin, 35*(3), 50-54.

5. Siemens, G., & Gasevic, D. (2012). Guest editorial-learning and knowledge analytics. *Educational Technology & Society, 15*(3), 1-2.

6. Wiliam, D., & Black, P. (1996). Meanings and consequences: A basis for distinguishing formative and summative functions of assessment? *British Educational Research Journal, 22*(5), 537-548.

7. Forehand, M. (2010). Bloom's taxonomy. Emerging Perspectives on Learning, Teaching, and Technology, , 41-47.

8. Aboulsoud, S. H. (2011). Formative versus summative assessment. *Education for Health (Abingdon, England), 24*(2), 651. doi:651 [pii]

9. Alisi, T. M., Del Bimbo, A., & Valli, A. (2005). Natural interfaces to enhance visitors' experiences. *Multimedia, IEEE, 12*(3), 80-85.

10. Malizia, A., & Bellucci, A. (2012). The artificiality of natural user interfaces. *Communications of the ACM, 55*(3), 36-38.

11. Fowles, D. (2005). Literature review on effects on assessment of e-marking.

12. Centers for Medicare & Medicaid Services (CMS) Office of Information Service (2008). *Selecting a development approach*. Webarticle. United States Department of Health and Human Services (HHS). Re-validated: March 27, 2008. Retrieved 27 Oct 2008

13. Pokkunuri, B. P. (1989). Object oriented programming. *ACM Sigplan Notices, 24*(11), 96-101.

14. Eder, J., Kappel, G., & Schrefl, M. (1994). Coupling and cohesion in object-oriented systems. *Technical Reprot, University of Klagenfurt, Austria,*

15. Hitz, M., & Montazeri, B. (1995). Measuring coupling and cohesion in object-oriented systems na.

16. Boone, H. N., & Boone, D. A. (2012). Analyzing likert data. *Journal of Extension, 50(2),* 1-5.

17. Android design principles | android developers Retrieved from http://developer.android.com/design/get-started/principles.html

18. Action bar | android developers Retrieved from http://developer.android.com/guide/topics/ui/actionbar.html

19. Baird, J. (1998). What's in a name? experiments with blind marking in A-level examinations. *Educational Research, 40*(2), 191-202.

20. Smith, R. (2007). An overview of the tesseract OCR engine. *Icdar, , 7* 629-633.

21. Dabbish, L., Stuart, C., Tsay, J., & Herbsleb, J. (2012). Social coding in GitHub: Transparency and collaboration in an open software repository. *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work,* 1277-1286.

9. Appendix A – User Testing Questionnaire

General

All participants must answer this section

1. **Which interface did you use?**
   Mark only one oval.

   ◯ Web
   ◯ Tablet

2. **Was the student's answer readable in terms of image quality (not handwriting)?**
   Mark only one oval.

   ◯ Yes
   ◯ No

3. **How complicated was it to mark using this interface?**
   Mark only one oval.

	1	2	3	4	5	
Very Difficult	◯	◯	◯	◯	◯	Very Easy

4. **How would you rate the effectiveness of the screen layout?**
   i.e. displaying the question, model answer and student answer at the same time
   Mark only one oval.

	1	2	3	4	5	
Very uneffective	◯	◯	◯	◯	◯	Very effective

5. **Would you say that the interface improved your marking speed?**
   Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

Web Interface

If you used the web interface during this session, please answer the following questions.

6. **How intuitive did you find it to mark with a mouse?**
   Mark only one oval.

	1	2	3	4	5	
Very Unintuitive	◯	◯	◯	◯	◯	Very Intuitive

7. **If you found it unintuitive, which aspect/s would you say need/s to be improved?**

Tablet Interface

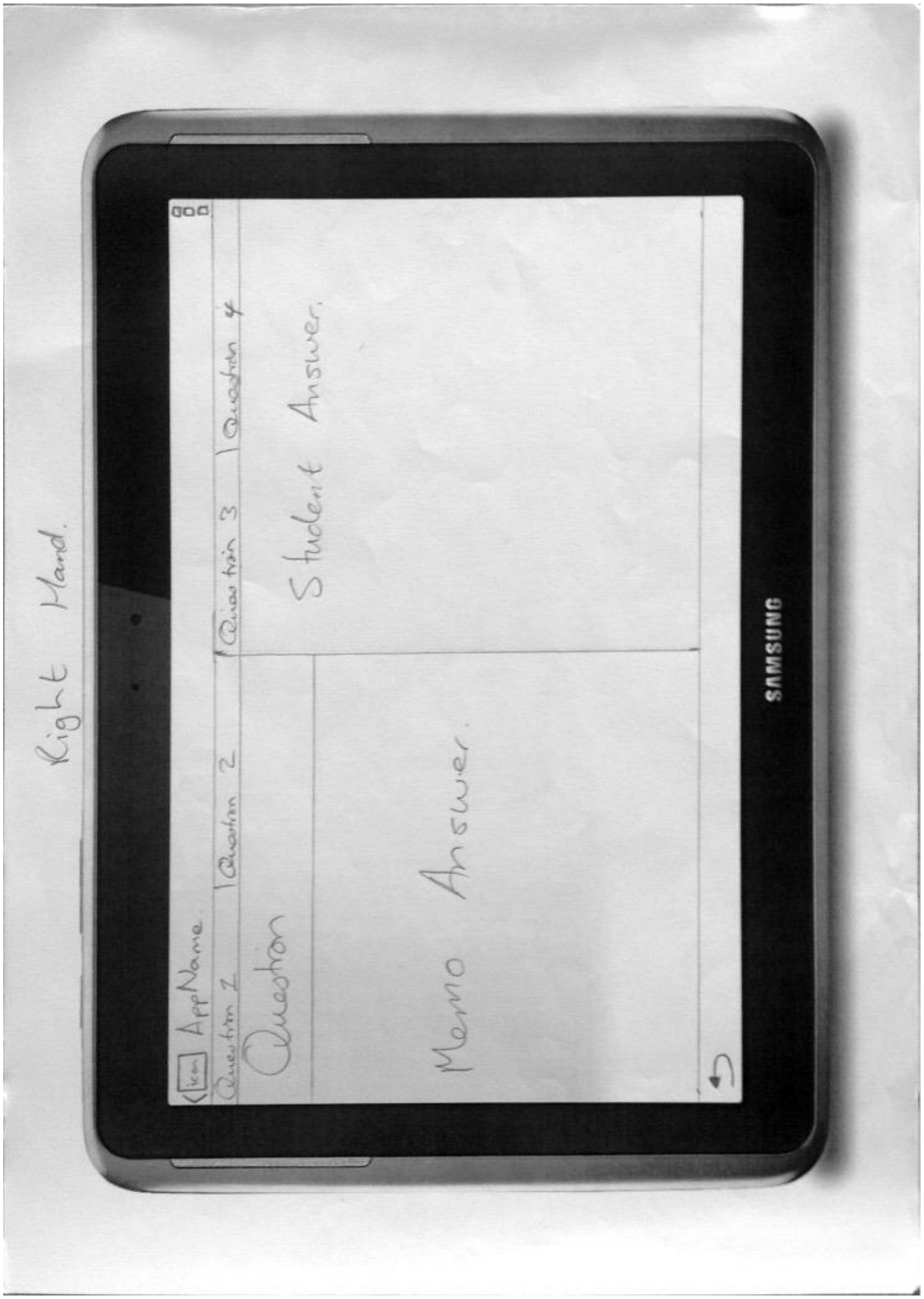
If you used the tablet interface during this session, please answer the following questions.

8. **How intuitive would you say was to use the stylus?**
   Mark only one oval.

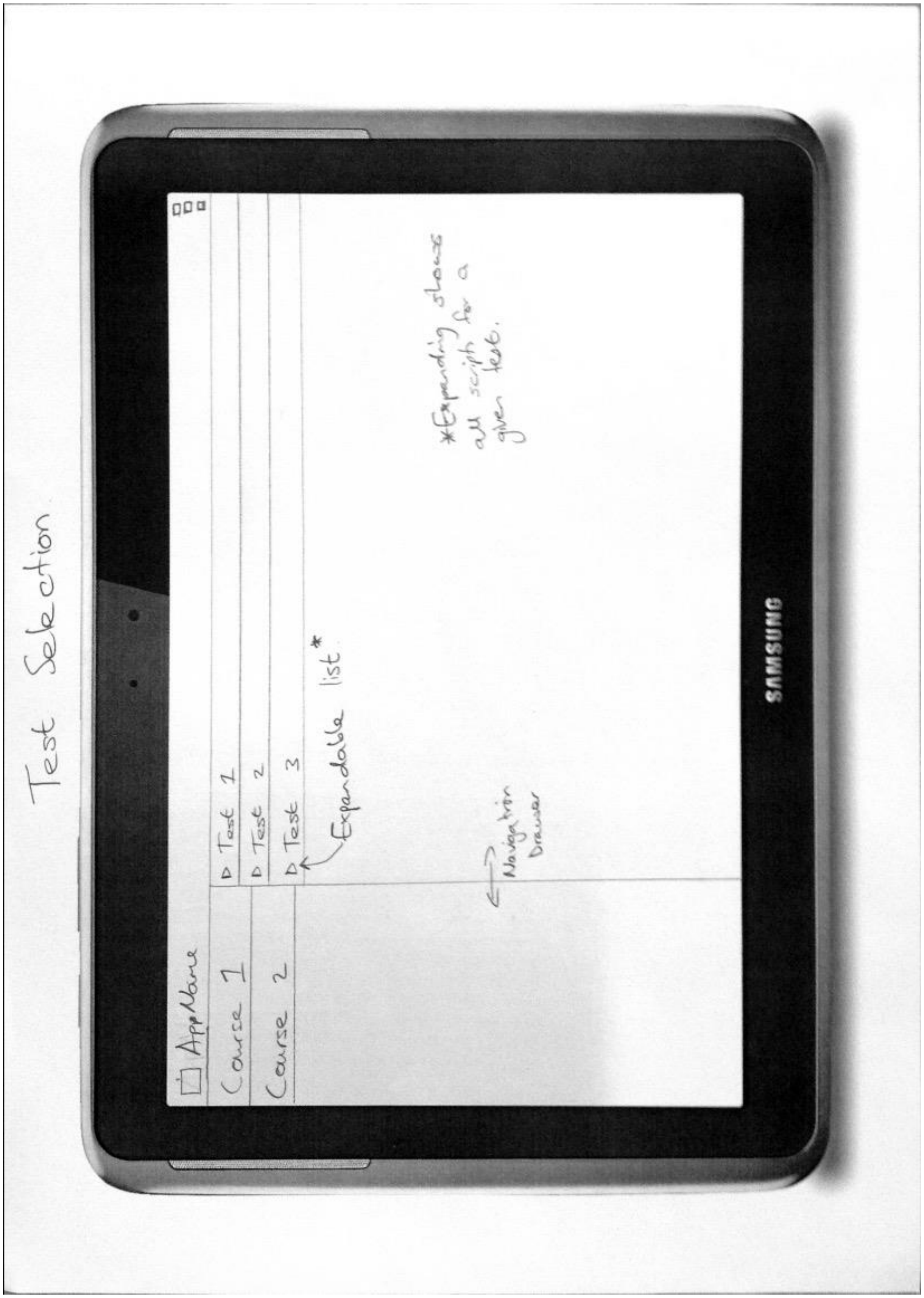
	1	2	3	4	5	
Very Unintuitive	◯	◯	◯	◯	◯	Very Intuitive

9. **If you found it unintuitive, which aspect/s would you say need/s to be improved?**

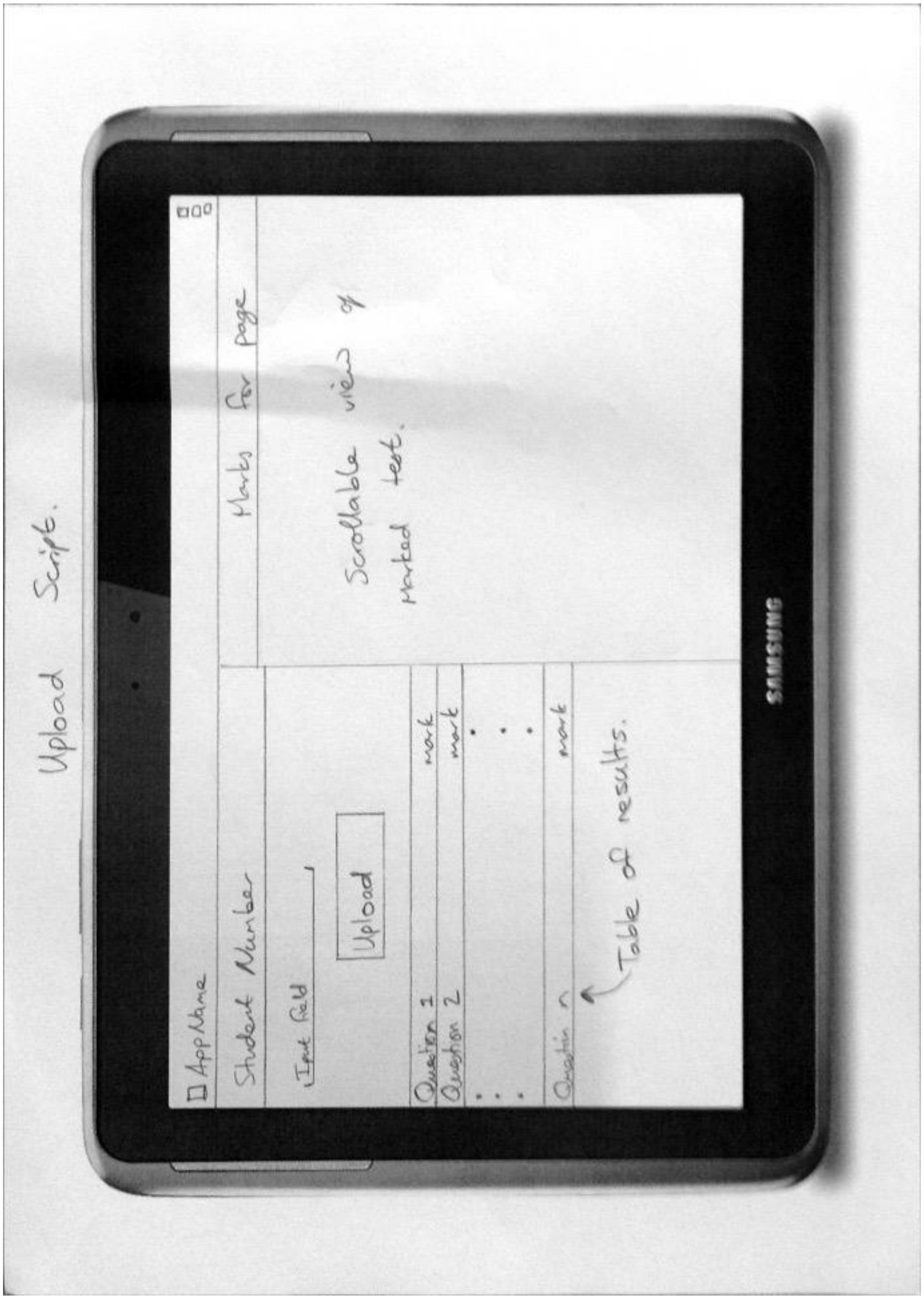
10. Appendix B.1 – Marking Screen Mockup



11. Appendix B.2 – Test Selection



12. Appendix B.3 – Final Screen



13. Appendix C

Faculty of Science Clearance

Faculty of Science
University of Cape Town
RONDEBOSCH 7701
South Africa

E-mail: richard.hill@uct.ac.za
Telephone: + 27 21 650 2786
Fax: + 27 21 650 3456

8 July 2014

Ms Z Mathews & Mr V Soudien

Department of Computer Sciences

Dear Ms Mathews & Mr Soudien

ELECTRONIC MARKING

I am pleased to inform you that the Faculty of Science Research Ethics Committee has approved the above-named application for research ethics clearance, subject to the conditions listed below. You are required to:

- implement the measures described in your application to ensure that the process of your research is ethically sound;

- uphold ethical principles throughout all stages of the research, responding appropriately to unanticipated issues: please contact me if you need advice on ethical issues that arise.

Your approval code is: FSREC 041– 2014

I wish you success in your research.

Yours sincerely

[illegible]

Dr Richard C Hill

Chair: Faculty of Science Research Ethics Committee

Access to Students Clearance

	RESEARCH ACCESS TO STUDENTS	**DSA 100**

NOTES

1. This form must be **FULLY** completed by the applicant/s who want to access UCT students for the purpose of research or surveys.
2. Return the fully completed **(a) DSA 100** application form **by email**, in **the same word format**, together with **your: (b) research proposal inclusive of your survey, (c) copy of your ethics approval letter / proof (d) informed consent letter** to:
   Moonira.Khan@uct.ac.za. You application will be attended to by the Executive Director, Department of Student Affairs (DSA), UCT.
3. The turnaround time for a reply is **approximately 10 working days.**
4. NB: It is the responsibility of the researcher/s to apply for and to obtain **ethics approval and to comply with amendments that may be requested;** as well as **to obtain** approval to access UCT staff and/or UCT students, from the following, respectively:
   (a) **Ethics:** Chairperson, Faculty Research Ethics Committee' (FREC) for ethics approval, (b) **Staff access:** Executive Director: HR for approval to access UCT staff, and (c) **Student access:** Executive Director: Student Affairs for approval to access UCT students.
5. **Note:** UCT Senate Research Protocols requires compliance to the above, even if prior approval has been obtained **from any other institution/agency. UCT's research protocol requirements applies to *all* persons, institutions and agencies from UCT and external to**
   UCT who want to conduct research for academic, marketing or service related reasons at UCT.

SECTION A: RESEARCH APPLICANT/S DETAILS

Position	Staff / Student No	Title and Name		Contact Details (Email / Cell / land line)
A.1 Student Number	MTHZAH002 SDNVIC001	Ms Zahraa Mathews Mr Victor Soudien		MTHZAH002@myuct.ac.za SDNVIC001@myuct.ac.za
A.2 Academic / PASS Staff No.				
A.3 Visitor/ Researcher ID No.				
A.4 University at which a student or employee		Address if ***not*** UCT:		
A.5 Faculty/ Department/School	Department of Computer Science			
A.6 APPLICANTS DETAILS If different from above	Title and Name		Tel.	Email

SECTION B: RESEARCHER/S SUPERVISOR/S DETAILS

Position	Title and Name	Tel.	Email
B.1 Supervisor	Gary Stewart		gstewart@cs.uct.ac.za
B.2 Co-Supervisor/s			

SECTION C: APPLICANT'S RESEARCH STUDY FIELD AND APPROVAL STATUS

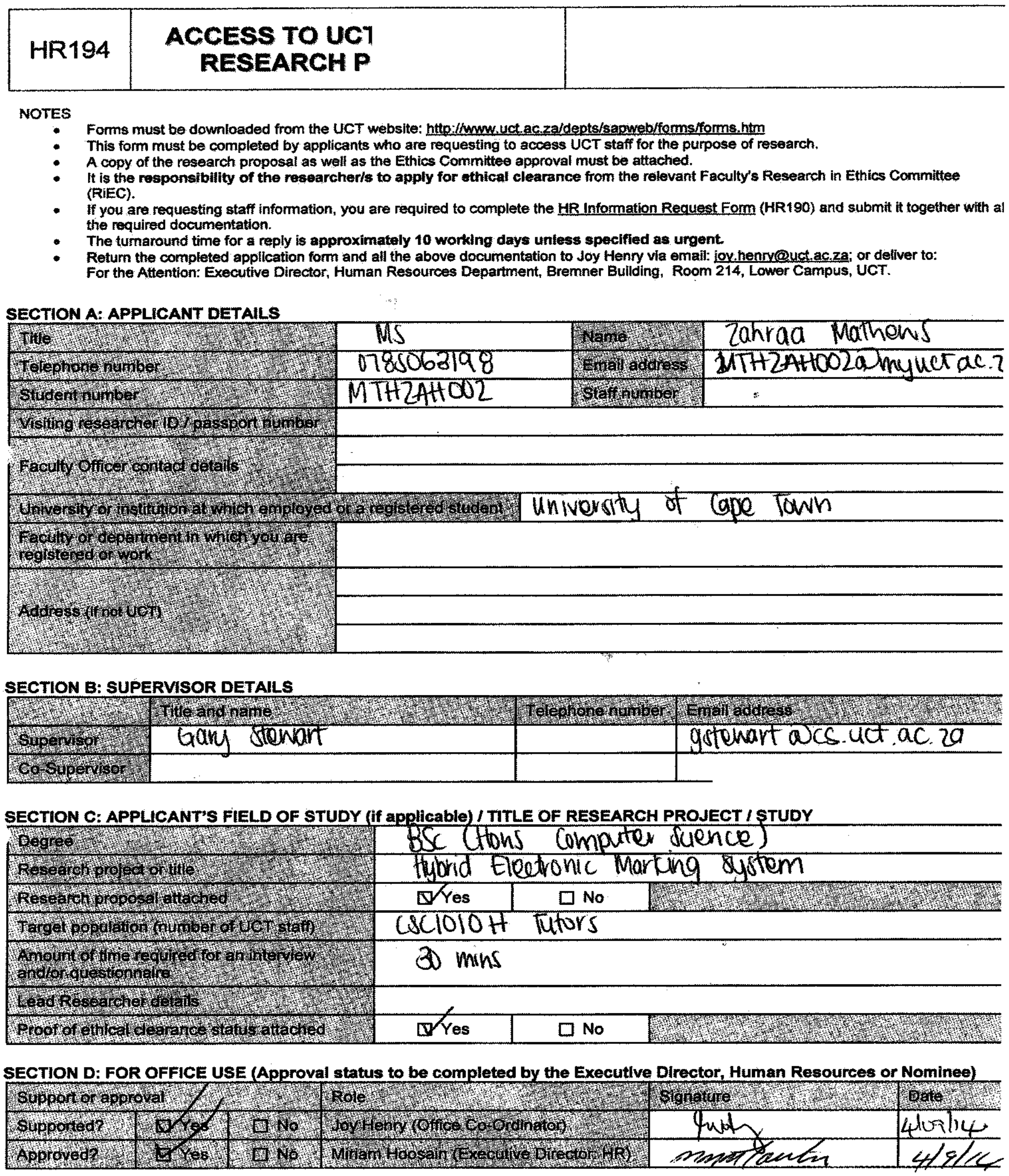
C.1 Degree (if a student)	**BSc (Honours) in Computer Science**
C.2 Research Project Title	
C.3 Research Proposal	**Attached:** **Yes** ■ **No** ☐
C.4 Target population	**Students registered for CSC1010H**
C.5 Lead Researcher details	**If different from applicant:**
C6. Will use research assistant/s	**Yes** ☐ **No** ■ **If yes- provide a list of names, contact details and ID no.**
C.7 Research Methodology and Informed consent:	**Research methodology**: Testing an electronic marking system for class tests through controlled access. **Informed consent:** Controlled access rights to markers of the electronic system
C.8 Ethics clearance status from UCT's Faculty Ethics Research Committee (FREC)	**Approved by the FREC** **Yes** ■ **With amendments:** **Yes** ■ **No** ☐ **Ref. No.: FSREC 041-2014** **(a) Attach copy of your ethics approval. Attached: Yes / No** **(b) State date and reference no. of ethics approval: Date: 8 July 2014**

SECTION D: APPLICANT/S APPROVAL STATUS FOR ACCESS TO STUDENTS FOR RESEARCH PURPOSE (*To be completed by the ED, DSA or Nominee)*

	Approved / With Terms / Not	*** Conditional approval with terms**	**Applicant/s Ref. No.:**
D.1 APPROVAL STATUS	***Yes - Approved*** ■ ~~**** Yes / No***~~	**(a)** Access to students for this research study must only be undertaken after written ethics approval has been obtained. **(b)** In event any ethics conditions are attached, these must be complied with before access to students.	**MTHZAH002 / Ms Zahraa Mathews** **SDNVIC001 /Mr Victor Soudien**

	Designation	**Name**	**Signature**	**Date**
D.2 APPROVED BY:	***Executive Director Department of Student Affairs***	***Dr Moonira Khan***	[illegible]	***24 July 2014***

Access to Staff Clearance



HR194	**ACCESS TO UCT RESEARCH P**	

NOTES

- Forms must be downloaded from the UCT website: http://www.uct.ac.za/depts/sapweb/forms/forms.htm
- This form must be completed by applicants who are requesting to access UCT staff for the purpose of research.
- A copy of the research proposal as well as the Ethics Committee approval must be attached.
- It is the **responsibility of the researcher/s to apply for ethical clearance** from the relevant Faculty's Research in Ethics Committee (RiEC).
- If you are requesting staff information, you are required to complete the HR Information Request Form (HR190) and submit it together with al the required documentation.
- The turnaround time for a reply is **approximately 10 working days unless specified as urgent.**
- Return the completed application form and all the above documentation to Joy Henry via email: joy.henry@uct.ac.za; or deliver to: For the Attention: Executive Director, Human Resources Department, Bremner Building, Room 214, Lower Campus, UCT.

SECTION A: APPLICANT DETAILS

Title	MS	Name	Zahraa Mathews
Telephone number	0785063198	Email address	MTHZAH002@myuct.ac.z
Student number	MTHZAH002	Staff number	
Visiting researcher ID / passport number			
Faculty Officer contact details			
University or institution at which employed or a registered student	University of Cape Town		
Faculty or department in which you are registered or work			
Address (if not UCT)			

SECTION B: SUPERVISOR DETAILS

	Title and name	Telephone number	Email address
Supervisor	Gary Stewart		gstewart@cs.uct.ac.za
Co-Supervisor			

SECTION C: APPLICANT'S FIELD OF STUDY (if applicable) / TITLE OF RESEARCH PROJECT / STUDY

Degree	BSc (Hons Computer Science)	
Research project or title	Hybrid Electronic Marking System	
Research proposal attached	[✓ Yes]	No
Target population (number of UCT staff)	CSC1010H Tutors	
Amount of time required for an interview and/or questionnaire	30 mins	
Lead Researcher details		
Proof of ethical clearance status attached	[✓ Yes]	No

SECTION D: FOR OFFICE USE (Approval status to be completed by the Executive Director, Human Resources or Nominee)

Support or approval			Role	Signature	Date
Supported?	[✓ Yes]	No	Joy Henry (Office Co-Ordinator)	[illegible]	4/09/14
Approved?	[✓ Yes]	No	Miriam Hoosain (Executive Director: HR)	[illegible]	4/9/14

Please note that this form was completed in the name of Zahraa Mathews, the other researcher involved in this project on behalf of both researchers.