

MapMyNotes

Final Report for CS39440 Major Project

Author: Ryan Gouldsmith (ryg1@aber.ac.uk)

Supervisor: Dr. Hannah Dee (hmd1@aber.ac.uk)

4th March 2016

Version: 1.0 (Draft)

This report was submitted as partial fulfilment of a BSc degree in
Computer Science (G401)

Department of Computer Science
Aberystwyth University
Aberystwyth
Ceredigion
SY23 3DB
Wales, UK

Declaration of originality

In signing below, I confirm that:

- This submission is my own work, except where clearly indicated.
- I understand that there are severe penalties for Unacceptable Academic Practice, which can lead to loss of marks or even the withholding of a degree.
- I have read the regulations on Unacceptable Academic Practice from the University's Academic Quality and Records Office (AQRO) and the relevant sections of the current Student Handbook of the Department of Computer Science.
- In submitting this work I understand and agree to abide by the University's regulations governing these issues.

Name

Date

Consent to share this work

In signing below, I hereby agree to this dissertation being made available to other students and academic staff of the Aberystwyth Computer Science Department.

Name

Date

Acknowledgements

I am grateful to...

I'd like to thank...

Abstract

Include an abstract for your project. This should be no more than 300 words.

CONTENTS

1 Background & Objectives	1
1.1 Background	1
1.1.1 Taxonomy of notes	1
1.1.2 Handwriting recognition	3
1.1.3 Similar systems	3
1.1.4 Motivation	5
1.2 Analysis	5
1.2.1 Parsing a note	6
1.2.2 An OCR tool	6
1.2.3 What to parse from the note	6
1.2.4 Structuring of notes	6
1.2.5 OCR for the authors handwriting	6
1.2.6 What platform is most suitable	7
1.2.7 What should the application do	7
1.2.8 Calendar integration	7
1.2.9 Objectives	7
1.2.10 Compromising with objectives	8
1.3 Process	8
1.3.1 Scrum overview	8
1.3.2 Adapted Scrum	9
1.3.3 Incorporated Extreme Programming	9
Appendices	11
A Third-Party Code and Libraries	12
B Ethics Submission	13
C Code Examples	14
3.1 Random Number Generator	14
Annotated Bibliography	17

LIST OF FIGURES

1.1 A taxonomy showing the structure and classification of different types of notes and what is contained in a note.	2
---	---

LIST OF TABLES

Chapter 1

Background & Objectives

1.1 Background

Handwriting notes is still considered to be an important aspect of note taking. Smoker et al. [16] conducted a study comparing handwritten text against digital text for memory retention and out of 61 adults 72.1% preferred to take notes using pen and paper, rather than on a computer. Smoker et al. concluded that recollection rates for handwritten text was greater than that of typed text proving that handwritten notes are better for a user's memory retention.

Technology has advanced and people are becoming more connected with distributed services through the cloud as well as tracking things in their life digitally; Google Calendar is an example of this. Therefore, there's a need to ensure that memory retention with handwritten notes is carried forward into the digital age.

1.1.1 Taxonomy of notes

When notes are made they will often be very different from any other note. Some are semi-structured and some are "back of the envelope" kind of notes. When thinking about an application to analyse notes, first there has to be consideration for what a note will consist of. A taxonomy, by definition, is a biological term for a classification of similar sections, showing how things are linked together [20].

Notes can be thought of as a collection of similar classifications, whether this is the pure textual descriptions of a note or whether this is purely pictorial form or a mixture of both. However, the notes are normally split into three distinct categories:

1. Textual descriptions
2. Diagrams
3. Graphs

In Figure 1.1 it shows a taxonomy of the different aspects which may form a part of a note. Textual descriptions form the core content of a note, this is essentially the important aspect that a note-taker is trying to remember and write down. Different note-takers form their notes in different

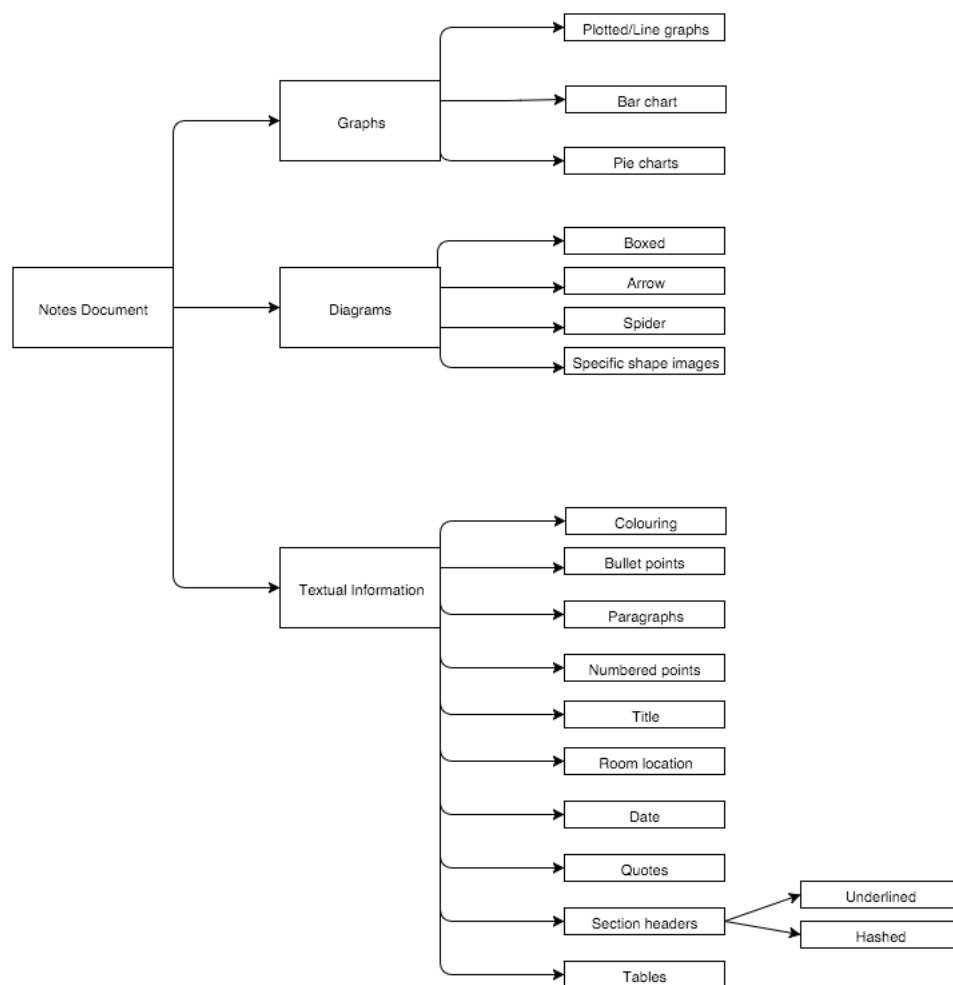


Figure 1.1: A taxonomy showing the structure and classification of different types of notes and what is contained in a note.

ways, for example the headings may be underlined or hashed - if they were adopting a mark-down style approach. These sections help to show that there's a break in the content, and should be sub-sectioned. Textual points that are short, but important, are often characterised by a colon or a bullet point; these are the most common form of concise note building, in the classification.

Coloured text is often used for a variety of reasons: it stands out on the page and improves memory retention of that text [3]. Both congruent and incongruent coloured text helped to increase memory retention of post-graduate learners [12]. With congruent text, Olurinola et al. showed that for 30 students with a 20 word list, a 10.9 mean retention rate was achieved and a 8.1 mean retention rate was achieved for incongruent text. The studies conducted show that coloured items would improve a user's retention rate in a lecture.

Finally, tables help to represent textual content in tabular form - this is often good in notes for comparisons.

Graphs are great visual tools for users to help to convey important textual information easily. Naturally, they have their limitations such as they come in different shapes and sizes, such as a line-graph, pie chart or a bar chart. Coupled with graphs, notes often consist of diagram drawings.

In Figure 1.1, there are different sections and classifications of a diagram: boxed, arrow etc. Each one has its own purpose and arrowed and box can overlap; UML diagrams are a case of this. Spider diagrams are probably the hardest to represent, due to the varying sizes and whether the user draws circles or clouds. Furthermore, specific shape diagrams are conceptually hard to think about as it depends on the domain in which the user is drawing the note. For example, a person in Biology may draw a stick person, whereas someone in Computer Science may draw a computer.

Identifying a taxonomy of notes is imperative when considering what to parse from a note as it helps to define a domain of possible classifications. By identifying the classifications it will acknowledge what sections can be parsed by a specific technique. For example, textural information and bullet point lists can be parsed via text-recognition however, for diagram recognition that would involve image manipulation.

1.1.2 Handwriting recognition

Analysing a user's handwriting is a complex process and one which requires lots of research. Handwriting recognition has had successes in the past from machine learning techniques, such as neural networks [7]. Knerr et al. yields a 10% rejection rate and a 1% error rate, with the use of Neural Networks on the U.S. Postal service data collection, showing that handwriting recognition is still very much an active research area, where solutions are still being developed to optimise the correct classifications of text.

Another approach is to analyse handwriting via an OCR (optical character recognition) tool. Rakshit et al. [14] used the open-source tool, Tesseract [18] to analyse Roman scripts. The system was trained on handwriting identified from those scripts. Rakshit et al. recorded an 83.5% accuracy on 1133 characters. It is noted in the paper that "over-segmentation" is a problem with the Tesseract engine.

Another issue to overcome when analysing handwriting is disjointed characters; this is where sections of the letter are split off from the main body of the character, for example, the letter i. Rakshit et al. concludes that this is an issue which is ever-present in the Tesseract engine, with around a 53% misclassification on this character alone.

The problem of handwriting recognition has not been solved - but tools such as Tesseract offers support for improvement in this field. As Rakshit et al., discusses training had to be conducted with Tesseract to ensure that it can identify handwriting successfully. As a result, every implementation of handwriting recognition needs significant amounts of data of varying quality, so that a system can succeed. However, considering Tesseract's high recognition rate of 83.5% experienced by Rakshit et al., it is a viable option to use for handwriting recognition.

1.1.3 Similar systems

With note-taking on digital devices becoming more widely available, there has become an influx in note-taking and organisational applications available for users. These are predominately WYSIWYG (what you see is what you get) editors - which allow a great deal of flexibility. When evaluating existing systems three were predominantly used and they were:

- OneNote

- EverNote
- Google Keep

1.1.3.1 OneNote

OneNote [9] is a note-taking and organisational application made by Microsoft, offering the functionality to add text, photos, drag and drop photos onto a plain canvas. In recent times, OneNote has developed functionality to analyse a user's handwriting, from say a stylus, and interpret the text they entered [11]. In OneNote you can insert a note into a document and then it would interpret the text from the note.

There is a wide range of product support from mobile based applications to web versions of their software. Office Lens [10] can be used in conjunction with the OneNote to help to take photos and automatically crop the image and then save them to OneNote. This feature is important and should be considered for the *MapMyNotes* application. The process requires the user to sign in with a Microsoft account. When creating notes, OneNote formats collated notes into a series of "notebooks".

One feature which was noted when analysing the system is automatic saving of the note, reducing the need for a user to click save. Additionally, when using OneNote it feels very much like Microsoft Word - with the similar layout that gives most users a similar user experience feel with its intuitive WYSIWYG editor.

1.1.3.2 EverNote

EverNote [4] is a note-taking and organisational application, it is supported as a web application, bespoke desktop application and a mobile application. EverNote is widely used and provides a wide range of functionality a user would need to digitise their notes.

EverNote have released development articles [1] stating that OCR recognition on images is possible. This would allow the user to upload an image outputting a list of potential words for each word found in the image. Like OneNote, the notes are collated into Notebooks, offering a WYSIWYG editor, giving the user full control of the content that is entered. When uploading an image to the web version it gives the option to edit the PDF and images, however it seems as though an additional application has to be downloaded, specific to the user's platform, to be able to utilise this functionality.

According to the website, it does do OCR recognition, however whilst using the web application there was no information regarding extracting of text from the application. Additionally, there seemed to be no way to save the note to a calendar item - only the option to send via a link.

1.1.3.3 Google Keep

Google Keep [5] is a note taking application produced by Google with mobile and website support. Google Keep allows a user to attach an image to their note, attempt to extract the text from an image and save this in the body of the note. In addition it allows the user to tag a title and add an associated body.

An important design feature that it does not offer is the support of a WYSIWG editor; a default text box has been preferred, offering a more raw feel to the application. They have the option of a “remind me” feature, which will get synced to their calendar as a reminder - but there’s no easy way to add it to a calendar event.

Google Keep seems as though it’s more suited for TODO lists and jotting down quick notes, rather than an archiving tool suitable for substantial note taking. Nevertheless, the tagging with labels is a nice feature and the filter by image is a smart tool; this only shows notes with specific images. The simplicity of the user interface and the ease in which text can be extract provides a great reference going forward.

1.1.3.4 Reflection on the systems

These three existing products are widely used by the every day note-taker. They have been developed to a high quality and give the user full control of what their notes can consist of. The automatically cropping of an image is an important feature and should be considered for the application going forward. *MapMyNotes* aims to try and give the user full control of their lecture notes content, so that they can find their notes easily.

MapMyNotes intends to differ from EverNote’s text extraction by providing a one to one comparison of the text, rather than a list of potential words.

After the analysis of the existing products there are certain aspects which would be regarded as necessary: a simple way to view the notes, a way to filter the notes and a simple UI which feels more like an application rather than a website.

1.1.4 Motivation

The author handwrites his notes during lectures and often are discarded in crowded notebooks until they are needed for an assignment or examination.

A calendar event is already stored for every lecture that the author goes to, so it would be useful if there was a way to associate each of the notes taken to that calendar event. This would ensure that all the information is located in one easy place that can be found again, instead of trawling through lots of paper and trying to find the content again. This would aid in reducing the chances of lost notes from paper slipping out of the notebook or pages being damaged due to rain or creases.

1.2 Analysis

As the project was originally proposed by Dr Harry Strange, a meeting was arranged to discuss the initial ideas that he wished the application would follow. It was here that it was highlighted that Dr Harry Strange wants to take a photo of his notes, archive them with specific data, make them searchable and integrate them with existing calendar entries he had for a given date.

1.2.1 Parsing a note

In conjunction with the information gathered a taxonomy of notes was collated, helping to deconstruct what a note consists of. Analysing the taxonomy produced a comprehensive breakdown of what could be parsed as text. After seeing that text formed a main component of a note the key efforts of the application would focus on parsing the text. Diagrams, graphs and images were considered when thinking of what should be extracted from an image, however this was placed as a task for the future. This required a task to investigate how to reliably extract the text from an image.

1.2.2 An OCR tool

Handwriting recognition has been an active research project for a while. There could have been the possibility of creating a bespoke handwriting recognition tool, using machine learning techniques, but that would distract from the actual problem which is this available tool to archive notes.

Therefore an OCR tool would have to be chosen to analyse the text. Choosing a sensible OCR tool with good recognition rates would be important - so a task was created to explore and look at possible solutions.

1.2.3 What to parse from the note

From research conducted into Google Keep it was clear that analysing the text would be a great aspect to include in the application. The real question is what should be parsed from the note? By looking at the overall structure of the application and what it entailed then it was agreed to just parse the note's associated meta-data: the title, lecturer, date and module code. Recalling that Google Keep parses all the text and EverNote gives a list of suggested words, it was decided that a tool would be developed to suggest the meta-data but it does not automatically tag the meta-data.

1.2.4 Structuring of notes

In conjunction with analysing what to parse, a sensible structure would have to be applied to notes used in the application. A task to create and find a good set of rules would have to be collated to ensure that notes could be parsed confidently. This reduced the complexity of incorporating natural language processing in the application, which would be implausible to be completed within the timeframe.

1.2.5 OCR for the authors handwriting

After research into OCR technologies, such as Tesseract [18], it was established that analysing handwriting is a complicated process. Instead of trying to train it on a lot of dummy data, it would be trained to recognise the author's handwriting. A task was created to train the user's handwriting data and this would run throughout the duration of the project.

1.2.6 What platform is most suitable

During the meeting with Dr Harry Strange one of the core features that was needed was for the application to be accessible regardless of where the user is. After the research was conducted all the aforementioned software tools have a web application version of their system. A mobile application was considered but only one version of the application would be made, either Android or iPhone, therefore preventing other phone users from using the application. A bespoke desktop application was considered for a long time, however, the user would have to ensure infrastructure decisions, such as databases, are correctly set up. As a result a web application was chosen - following research found; the next steps were to consider appropriate tools to use.

1.2.7 What should the application do

From analysing all three of the chosen research systems, it was clearly identifiable that they all have the ability the view all notes, searching, deleting, adding and editing a note. Taking these ideas on-board, they were set as a high-level task and something that the core system *must* do.

Reflecting on the premise of the application, that it was to aid the organisation of lecture notes, it was concluded that the best way to search for notes would be by module code, as most University students would want to find specific module notes. This created the high level task that notes must be searchable by their module code.

1.2.8 Calendar integration

From evaluating the systems it was noticed that there was not a clear way to integrate into a calendar. Reflecting on the conversations with Dr Harry Strange, integrating with the calendar was important for keeping the different systems together. From an AYTm survey, in December 2015, [13] Google calendar is the most popular calendar application, therefore due to time constraints Google calendar was the choice of integration and other competitors such as Microsoft would not be implemented. This formulates the task of integrating the calendar into the application to save the url of the note to a specific event.

1.2.9 Objectives

As a result of the analysis of the problem, the following high-level requirements were formulated:

1. Investigate how to extract handwriting text from an image - this will involve looking into ways OCR tools can interpret handwriting.
2. Train the OCR to recognise text of the author's handwriting.
3. Produce a set of a rules which a note must comply to.
4. Produce a web application to form the core part of the product. This includes allowing a user to upload an image, display the image. Add appropriate tagging to a note such as module code.

5. The user must be able to search for a given module code, showing the full list of notes based on the module code entered.
6. The backend of the application must conduct basic OCR recognition, analysing the first 3 lines of the notes.
7. The backend must integrate with a calendar to archive the notes away later to be found again.

1.2.10 Compromising with objectives

Some additional compromises were made in-light of the analysis due to the complexity of the tasks at hand.

- It would be nice to have image extraction from a note and incorporating a WYSIWYG editor into the application, like OneNote.
- Full OCR on all the characters. This would then output the text to a blank canvas.
- Make the handwriting training generic enough to identify a wide range of users handwriting.

It is worth noting down that the project supervisor Dr Hannah Dee felt as though the handwriting training would be too much for the dissertation and should be done as a “maybe”. After much deliberation it was decided to include it, but as a background process.

1.3 Process

Software projects often have a degree of uncertainty with requirements at the beginning, these projects lend themselves to an Agile approach. Whereas more structured applications with requirements which are well known are suited to a plan-driven approach.

For this project there are a lot of tasks which are not 100% definable at the start of the project. In addition to this certain tasks, such as training the author’s handwriting data, can not be truly estimated down to a fixed time. Often new requirements would emerge from weekly meetings and only high level requirements were in-place from the start of the project. As a result, a plan-driven approach such as the Waterfall model would not be appropriate, and an Agile methodology was implemented.

1.3.1 Scrum overview

Scrum [15] is a methodology used by teams to improve productivity where possible. Due to this being a single person project, a Scrum approach has to be modified. Sprints are set time-boxes where tasks are completed. These vary from one to four weeks in length but a shorter sprint means the developer can act on quicker feedback.

Scrum organises its work into user stories to ensure client valued work is being completed. They are normally collected at the start of the project and put into the backlog, which is a collection of client valued work. At the start of each sprint user stories are selected from the backlog with an estimation on complexity performed. Finally, at the end of the sprint a review and retrospective is conducted to analyse the sprint, identifying what went well and what could be improved.

1.3.2 Adapted Scrum

During the project this methodology was embraced and adapted. A one week long sprint was adopted which coincided with a weekly supervisor meeting. Epics (a high level version of a story) were identified at the start of the project to reflect the work completed in the analysis phase.

The epic was then broken down into user stories. Each user story was formulated as: “As a <role> I want to <feature> so that <resolution>”. This gave specific client value that was known to have a purpose. Each of these stories were estimated on their complexity and compared to a “goldilock” task ¹.

For planning a sprint, the planning poker [17] technique was adopted; user-stories are estimated on a scale of 1, 2, 3, 5, 8 etc. When a task was estimated about 15 story points, it would be reflected upon to ensure the scope was fully understood - this would be broken down to sub-stories where appropriate.

At the end of a sprint a review and retrospective was conducted in the form of a blog post [6], instead of in a team. The retrospective was used to analyse what was achieved in the sprint, what went well and what needed to be improved upon. During this time, pre-planning was conducted to formulate a series of tasks to complete in the next sprint; this was agreed by the customer (Dr Hannah Dee).

Communication with the project supervisor was key to determine what needed to be completed. It was discussed if what was suggested was achievable in that weeks sprint, based on the total story points completed in the previous sprint; if 20 story points were completed in sprint 3 then 20 story points were estimated for sprint 4 - associated user stories were brought forward.

The project was managed on the open source management tool Taiga.io [19] which was invaluable, and provided built in functionality such as burndown charts per sprint. This shows how well story points are being completed and are used as an analytical tool for how well progression was being made.

Daily stand-ups were informally conducted with a peer. Cut from the usual 15 minutes to around 5 minutes, the conversation helped to identify if there were any issues, what had been completed yesterday and what would be completed that day. This provided a good way to analyse what needed to be achieved and keep in perspective how the sprint was going.

1.3.3 Incorporated Extreme Programming

In tandem with Scrum, Extreme programming [2] principles were integrated into the development process; merciless refactoring, continuous integration and test-driven development were borrowed from its principles.

1.3.3.1 Test-driven development

Test-driven development (TDD) is the process of writing tests prior to the implemented code. This allows the developer to think about the design prior to its implementation and can form part of the documentation [8]. This was implemented throughout the project, with both unit and acceptance

¹ A task which all other tasks are evaluated against.

tests being written before the code implementation.

TDD follows three cycles: red, green, refactor. Initially the test fails, then it passes then refactoring is performed to keep the simplest system.

1.3.3.2 Continuous Integration

Continuous Integration tools were a core part of the process in this project. Typically used to ensure that code is checked into a repository it was used to ensure that the application could be built in an isolated environment and pass all the tests.

1.3.3.3 CRC cards

Class, responsibilities and collaboration (CRC) cards [21] were used during the design section to consider how different classes were to be created and the responsibilities they share. This principle from Extreme Programming helped to keep the design simple and not convoluted.

Appendices

Appendix A

Third-Party Code and Libraries

If you have made use of any third party code or software libraries, i.e. any code that you have not designed and written yourself, then you must include this appendix.

As has been said in lectures, it is acceptable and likely that you will make use of third-party code and software libraries. The key requirement is that we understand what is your original work and what work is based on that of other people.

Therefore, you need to clearly state what you have used and where the original material can be found. Also, if you have made any changes to the original versions, you must explain what you have changed.

As an example, you might include a definition such as:

Apache POI library The project has been used to read and write Microsoft Excel files (XLS) as part of the interaction with the clients existing system for processing data. Version 3.10-FINAL was used. The library is open source and it is available from the Apache Software Foundation [?]. The library is released using the Apache License [?]. This library was used without modification.

Appendix B

Ethics Submission

This appendix includes a copy of the ethics submission for the project. After you have completed your Ethics submission, you will receive a PDF with a summary of the comments. That document should be embedded in this report, either as images, an embedded PDF or as copied text. The content should also include the Ethics Application Number that you receive.

Appendix C

Code Examples

3.1 Random Number Generator

The Bayes Durham Shuffle ensures that the psuedo random numbers used in the simulation are further shuffled, ensuring minimal correlation between subsequent random outputs [?].

```
#define IM1 2147483563
#define IM2 2147483399
#define AM (1.0/IM1)
#define IMM1 (IM1-1)
#define IA1 40014
#define IA2 40692
#define IQ1 53668
#define IQ2 52774
#define IR1 12211
#define IR2 3791
#define NTAB 32
#define NDIV (1+IMM1/NTAB)
#define EPS 1.2e-7
#define RNMIX (1.0 - EPS)

double ran2(long *idum)
{
    /*-----*/
    /* Minimum Standard Random Number Generator */
    /* Taken from Numerical recipies in C */
    /* Based on Park and Miller with Bays Durham Shuffle */
    /* Coupled Schrage methods for extra periodicity */
    /* Always call with negative number to initialise */
    /*-----*/

    int j;
    long k;
    static long idum2=123456789;
```

```
static long iy=0;
static long iv[NTAB];
double temp;

if (*idum <=0)
{
    if (-(*idum) < 1)
    {
        *idum = 1;
    }else
    {
        *idum = -(*idum);
    }
    idum2=(*idum);
    for (j=NTAB+7; j>=0; j--)
    {
        k = (*idum)/IQ1;
        *idum = IA1 *(*idum-k*IQ1) - IR1*k;
        if (*idum < 0)
        {
            *idum += IM1;
        }
        if (j < NTAB)
        {
            iv[j] = *idum;
        }
    }
    iy = iv[0];
}
k = (*idum)/IQ1;
*idum = IA1*(*idum-k*IQ1) - IR1*k;
if (*idum < 0)
{
    *idum += IM1;
}
k = (idum2)/IQ2;
idum2 = IA2*(idum2-k*IQ2) - IR2*k;
if (idum2 < 0)
{
    idum2 += IM2;
}
j = iy/NDIV;
iy=iv[j] - idum2;
iv[j] = *idum;
if (iy < 1)
{
    iy += IMM1;
}
```

```
if ((temp=AM*iy) > RNMx)
{
    return RNMx;
}else
{
    return temp;
}
}
```

Annotated Bibliography

- [1] “Evernote Tech Blog — The Care and Feeding of Elephants,” <https://blog.evernote.com/tech/2013/07/18/how-evernotes-image-recognition-works/>, 2013, last checked 25th March 2016.

An article explaining how Evernote does character recognition on images

- [2] R. Agarwal and D. Umphress, “Extreme Programming for a Single Person Team,” in *Proceedings of the 46th Annual Southeast Regional Conference on XX*, ser. ACM-SE 46. New York, NY, USA: ACM, 2008, pp. 82–87. [Online]. Available: <http://dx.doi.org/10.1145/1593105.1593127>

This paper was useful on how Extreme Programming can be modified to a single person project. It provided thought on the methodology which should be undertaken on the project and how different aspects of Extreme Programming can be used.

- [3] M. A. A. Dzulkifli and M. F. F. Mustafar, “The influence of colour on memory performance: a review.” *The Malaysian journal of medical sciences : MJMS*, vol. 20, no. 2, pp. 3–9, Mar. 2013. [Online]. Available: <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3743993/>

A paper reviewing whether colour helps with memory retention. Used for the analysis and further confirmation in the taxonomy of notes section.

- [4] Evernote, “The note-taking space for your life’s work — Evernote,” <https://evernote.com/?var=c>, 2016, last checked 17th April 2016.

The Evernote application is an example of the organisational and note-taking application that this project is looking at as a similar system.

- [5] Google, “Meet Google Keep, Save your thoughts, wherever you are - Keep Google,” <https://www.google.com/keep/>, 2016, last checked 17th April 2016.

Google keep is an organisational and note-taking application, it is used as part of the evaluation and background analysis. It was compared against what the application could do.

- [6] R. Gouldsmith, “Ryan Gouldsmith’s Blog,” <https://ryangouldsmith.uk/>, 2016, last checked TODO.

A collection of blog posts which explain the progress every week through a review and reflection post.

- [7] S. Knerr, L. Personnaz, and G. Dreyfus, “Handwritten digit recognition by neural networks with single-layer training,” *IEEE Transactions on Neural Networks*, vol. 3, no. 6, pp. 962–968, Nov. 1992. [Online]. Available: <http://dx.doi.org/10.1109/72.165597>

A paper describing how a Neural network was build to identify handwritten characters on the European database and the U.S. postal service database.

- [8] C. Maiden, “An Introduction to Test Driven Development — Code Enigma,” <https://www.codeenigma.com/community/blog/introduction-test-driven-development>, 2013, last checked 17th April 2016.

A blog post giving a detailed description of what Test-driven development includes. Gives supportive detail to discussing that tests can be viewed as documentation.

- [9] Microsoft, “Microsoft OneNote — The digital note-taking app for your devices,” <https://www.onenote.com/>, 2016, last checked 13 April 2016.

Used to look at and compare how similar note taking applications structure their application. Used the application to test the user interface and what functionality OneNote offered that may be useful for the application

- [10] —, “Office Lens Windows Apps on Microsoft Store,” <https://www.microsoft.com/en-gb/store/apps/office-lens/9wzdncrfj3t8>, 2016, last checked 17th April 2016.

The Microsoft Lens application which would automatically crop, resize and correctly orientate an image taken at an angle.

- [11] —, “Take handwritten notes in OneNote 2016 for Windows - OneNote,” <https://support.office.com/en-us/article/Take-handwritten-notes-in-OneNote-2016-for-Windows-0ec88c54-05f3-4cac-b452-9ee62cebbd4c>, 2016, last checked 17th April 2016.

An article on OneNote’s use of handwriting extraction from an image. Shows simply how to extract text from a given image.

- [12] O. Olurinola and O. Tayo, “Colour in learning: Its effect on the retention rate of graduate students,” *Journal of Education and Practice*, vol. 6, no. 14, p. 15, 2015.

Discusses a study which shows that coloured text is better for the memory retention rates, than that of non-coloured text. Used during the taxonomy of notes section.

- [13] A. Pilon, “Calendar Apps Stats: Google Calendar Named Most Popular — AYTM,” <https://aytm.com/blog/daily-survey-results/calendar-apps-survey/>, 2015, last checked 13th April 2016.

A survey showing that Google calendar was ranked the most used calendar people use. Added to the analysis stage to justify why Google calendar was chosen instead of other calendars available.

- [14] S. Rakshit and S. Basu, “Recognition of Handwritten Roman Script Using Tesseract Open source OCR Engine,” Mar. 2010. [Online]. Available: <http://arxiv.org/abs/1003.5891>

The paper presents a case-study into the use of the Tesseract OCR engine. It analyses how to use train the data on handwriting based recognition, drawing conclusions on where it's useful - as well as it's downfalls.

- [15] Scrum.org, "Resources — Scrum.org - The home of Scrum," <https://www.scrum.org/Resources>, 2016, last checked 17th April 2016.

The website for the scrum methodology principles. The website was used to reference the process and methodology which was adapted in the project

- [16] T. J. Smoker, C. E. Murphy, and A. K. Rockwell, "Comparing Memory for Handwriting versus Typing," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 53, no. 22, pp. 1744–1747, Oct. 2009. [Online]. Available: <http://dx.doi.org/10.1177/154193120905302218>

Used to show that there handwriting is still an important part of memory retention with note taking, compared to digital text

- [17] M. G. Software, "Planning Poker: Agile Estimating Made Easy," <https://www.mountaingoatsoftware.com/tools/planning-poker>, 2016, last checked 17th April 2016.

Showing the use of planning poker with exactly how it was implemented in the application using the scrum based approach.

- [18] Tesseract, "Tesseract Open Source OCR Engine," <https://github.com/tesseract-ocr/tesseract>, 2016, last checked 17th April 2016.

The open source optical character recognition engine which will be used in the application to analyse characters on a page.

- [19] Tiaga, "Taiga.io," <https://taiga.io/>, 2016, last checked TODO.

The project management tool which was utilised to help to keep track of the project's progress throughout the process. Utilised the Scrum tools available that the application gives.

- [20] M. Webster, "Taxonomy — Definition of Taxonomy by Merriam-Webster," <http://www.merriam-webster.com/dictionary/taxonomy>, 2016, last checked 17th April 2016.

A definition of exactly what a taxonomy is. Clearly labelling it as a classification of a problem.

- [21] D. Wells, "CRC Cards," <http://www.extremeprogramming.org/rules/crccards.html>, 1999, last checked 17th April 2016.

A description of what CRC cards are and why they're useful when considering the design of an application. Used as a reference material throughout the process, as well as during the chapter discussing the process.