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

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Implementation

1.1 Image processing

The image processing would prove to be an integral part of the application, enhancing the ability to learn handwriting more effectively and efficiently. The end script was implemented over a series of sprints.

1.1.1 Optimising Tesseract

After the design considerations to use OpenCV was chosen, specific algorithms would need to be implemented. After a discussion with Dr Hannah Dee, it was suggested that investigations into different binarisation scripts would be useful.

1.1.1.1 OTSU

[CITE CITE, Lab brooks page]

OTSU, created by Nobuyuki Otsu, is a binarisation technique which essentially converts an image to black and white. Otsu is a global thresholding algorithm, using the whole image as a comparsion. This is unlike local thresholding algorithms where comparisons are made pixel by pixel.CITE Automatic thresholding for defect detection].

Due to notes having non-uniform lighting by the intrusion of shadows on an image, this makes OTSU difficult to reliably binarise the image.

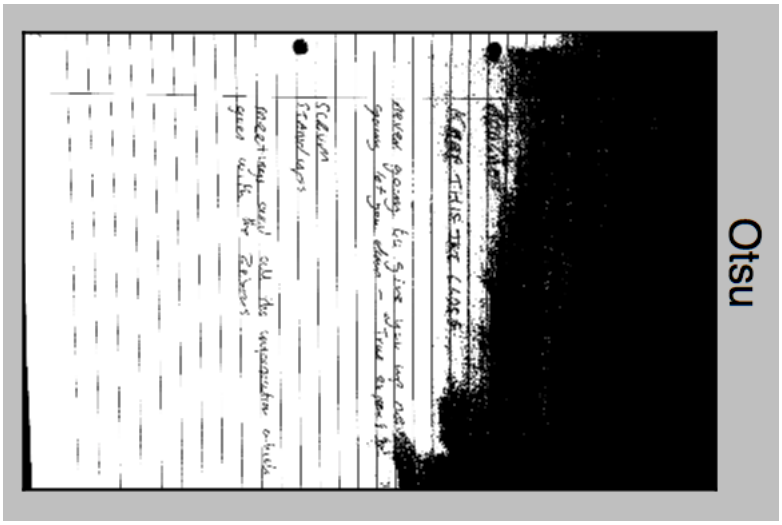

Figure 1.1: The use of OTSU binarisation technique on an image with a little shadow across the image

Figure 1.1.1.1 shows the binarisation technique on an image with the a slight shadow imposed on the image. Clearly it can be seen that it binarises the wrong sections of the image, this would be due to global thresholding.

The basic premise of OTSU is the image's grey-level values are segmented into a series of histograms. OTSU determines the optimial threshold value by "maximising the discriminant measure"[CITE OTSU PAPTER][CITE OPENCV]. In other words, OTSU attempts to maximise the margin between the histograms. From the maximising of the histograms, pixels can be segmented into their background or foreground pixels. [CITE LAB BROOKS PAGE].

HP, who created Tesseract [CITE], describe OTSU as its underlying pre-processing algorithm when it tries to identify characters. From the iterative spike work with OTSU it is clear to see, just from Figure 1.1.1.1, how Tesseract would be unable to clearly identify characters from that image.

Overall, OTSU, although it is a very solid binarisation method, it suffers from imposed shadows over images. This would not be the best option to choose when choosing an appropriate binarisation technique.

1.1.1.2 Adaptive Threshold

From the enlighting analysis of OTSU it was realised that using an OTSU thresholding ontop of an OTSU threshold from the Tesseract engine would not be beneficial. Therefore, in the next iteration of the script, an adaptive threshold approach was chosen.

Adaptive threshold will calculate the threshold over a series of smaller segments of the image [CITE]. This reduces the impact of shadows over an image, and does not consider global illumination as the key.

Using the OpenCV library there was two options [CITE OPEN CV]:

1. Gaussian adaptive threshold which is the weighted sum of the neighborhood

2. Mean adaptive threshold, which is the mean of neighborhood.

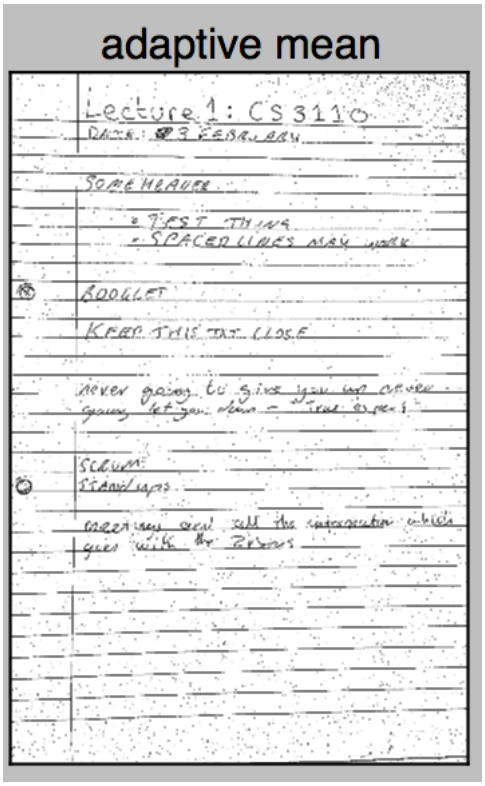

Figure 1.2: Adaptive mean threshold algorithm on a note, showing binarisation but there is still noise in the image.

The mean neighboured takes a block size around the pixel, say 4, and will work out the mean pixel value from that block. The mean value selected will then be selected as the thresholding value, which will determine whether pixels are background or foreground. [CITE] Figure 1.2 shows a mean adaptive threshold. Due to smoothing issues there is still noise on the image.

The gaussian operation differs from the mean as it uses a gaussian value over the sub-image. Firstly each "blocksize" is a value which surround the pixel. A default gaussian weight is then calculated [ADD calculation] based on the blocksize. For every pixel in the block, multiply it by the gaussian, an average weight is then taken and used as a threshold. [CITE LEARN CV][CITE OPEN CV].

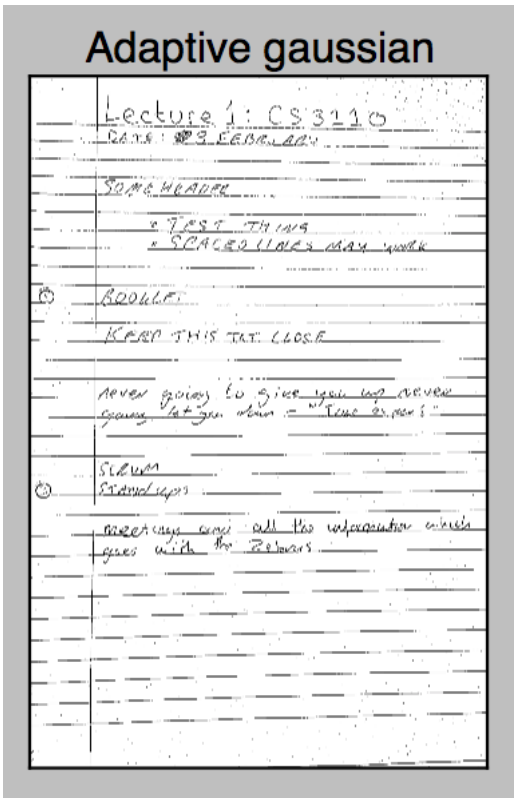

Figure 1.3: Adaptive Gaussian used over the image, showing a lot smoother of an image

Figure 1.3 shows the adaptive gaussian shows an image which is working it's way to binarisation. It does not have a shadow overlaying the image and the text, lines and little noise have been extracted.

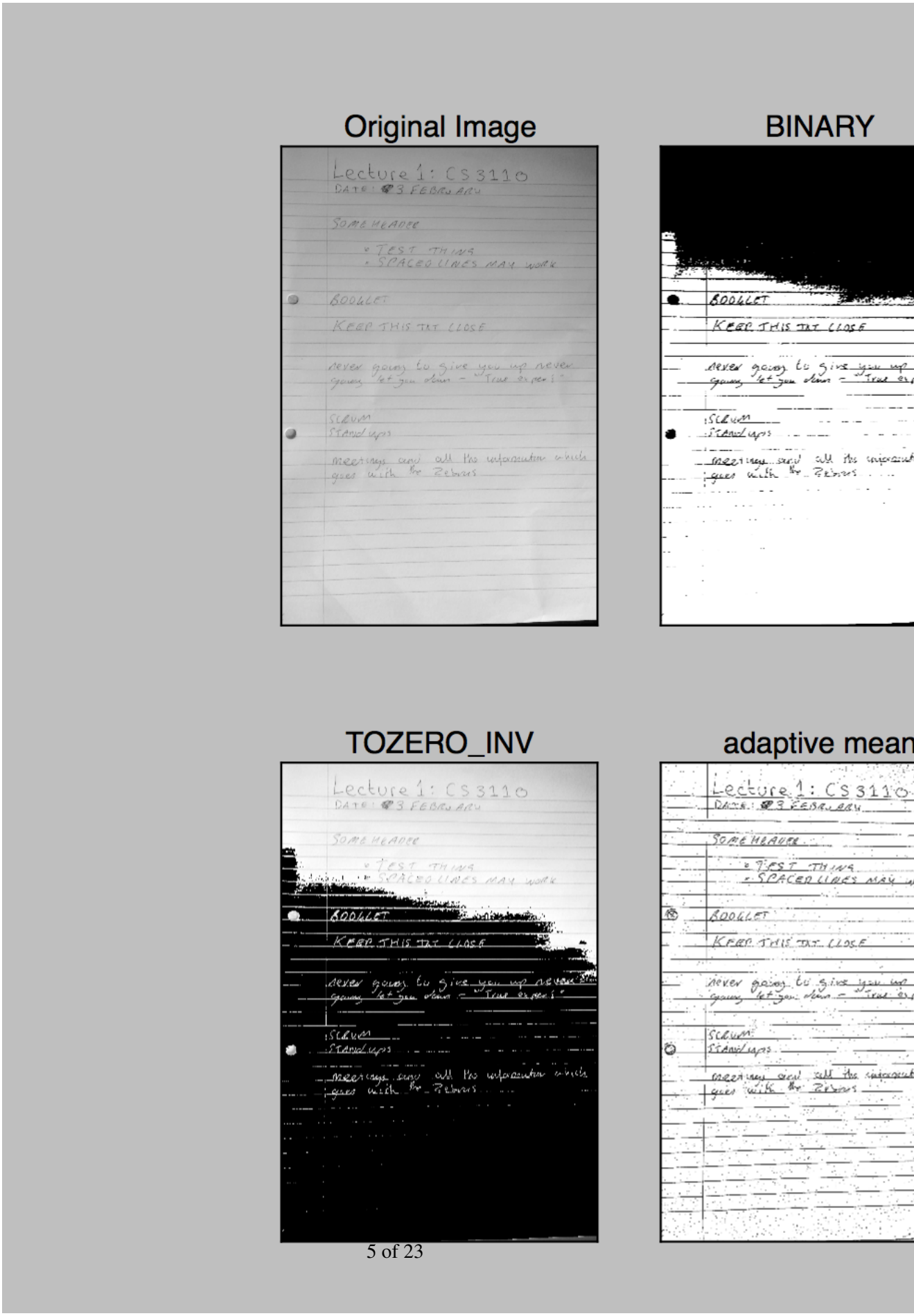

Figure 1.4: A variety of thresholding techniques used on the same note, showing adaptive threshold resulting in the best

Figure **??** displays the other types of thresholding options iteratively tried. It was decided that the gaussian adaptive thesholding would be implemented and improved with different morphological operations.

1.2 Lined paper

Intially standard lined paper was used for the notes, but the noise produced from the lines was too much to ensure reliable readings from, Tesseract. Refer to appendix [**?**] section [**?**].

1.2.1 Filtering the blue lines

Custom lined paper, with equal spacing was constructed to overcome this as discussed in section **??**.

Over a few iterations the main aim was to remove the blue lines from the image.

Algorithm 1 Initial removing the blue lines algorithm

1: **function** ((r)emove_lines)
2: $image \leftarrow read\_image\_as\_grayscale()$
3: $lower_black \leftarrow np.array([0, 0, 0])$
4: $upper_black \leftarrow np.array([175, 20, 95])$
5: $mask_black \leftarrow cv2.inRange(erode, lower\_black, upper\_black)$
6: $mask[np.where(mask\_black == 0)] \leftarrow 255$
7: **end function**

This process went through a series of iterations to try and overcome issues identified through each iteration. The first process was to extract all the values on the page which fell between a predefined grey-black range. However, this had its obvious problems that it would extract some of the line where a dark blue would contain black-like elements of colour.

Morphological operations such as erroding [CITE] and dilation [CITE] was performed on the image, but to no avail - there was still noice in the image which causing Tesseract to interpret them incorrectly. This also had an unwanted side-effect: the binarised characters would then become so undistinguishable that they would not be able to be located correctly by the OCR tool. They became so pixelated that even from a human eye it was hard to identify their meaning.

1.2.2 Only extracting the text

The following optimisation would identify the lines, but only extract the text that was needed. [CITE EXAMPLE ON OPEN CV] A gaussian adaptive threshold operation was performed over a median blurred, grayscale version of the note; successfully binarising both the text and the lines. The horizontal lines on the page were extracted using OpenCV's structuring element, $MORPH\_RECT$ [CITE].

Further errosion and dilation was applied to remove the black lines and any noise residue from the extraction. A series of intermediate blank image masks were created to transfer the text from

the image across to the mask. Although this carried all the text over, further line noise was also transferred across.

This suffered similar problems are the previous iteration, until connected components, via contours was discovered. Due to the morphological operations on the horizontal lines the connectivity between the pixels was very small. Thereby, choosing to use connected components would be able to extract the text from the image with little interference from the horizontal line noise. These connected components were then transferred to the new mask - only carrying the text from the image, not the lines.

Final morphological operations were included to improve the image quality. Overall the binarisation script works well. It can take a photo of an image in a terrible light source Fig x, for example, and it would succesfully output the binarised image which would have distinguishable text, coinciding with little noise on the image. The noise on the image has little interference with the Tesseract OCR and it yields better results than greyscale images. [CITE THE OPEN CV LYRIC EXAMPLE]

1.3 Handwriting Training

During the start of the handwriting training phase problems occurred such as it not reading the characters from a greyscale image correctly.

1.3.1 Training process

When using training data to be worked with Tesseract, it requires it to be in a specific file format. Such as: $< lang > . < font > .exp < number > .tiff$. When ran through a language, Tesseract outputs a box file which contains on each line: the character and the coordinates of this box. Trying to analyse this box was almost impossible. On the Tesseract wiki page [CITE] there's a link to jTessBoxEditor [CITE]. This tool was used to tag the boxes with their associated content.

Whilst using this editor the boxes could be expanded or shrunk to give the best possible fit to the characters. This was often utilised due to erroneous characters picked up by the editor.

There were a few issues when training the handwriting data, sometimes the data would not even be recongised.

1.4 Web application

1.4.1 OAuth

Working with the Google OAuth prooved to be quite tricky in some places. Using the Google client library to handle the OAuth2 interactions with the Google API's allowed for a reliable connection and exchanging of tokens.

During the interaction with the Google API's there was a time in which the API client failed and threw a random error. Confused as the service was working the prior day, an issue was made on their GitHub repository [CITE]. This issue miraculously dispearred after a clear of a cache

from the API tool.

One issue to consider when dealing with OAuth is handling the refresh tokens. When a user authenticated, the credentials were stored in the user session. In the response from the Google OAuth is a refresh token - this has an expiration date. Prior to realising that the client had a check to see if the expiration token expired, the user would be presented with an error informing them they have an invalid token.

The new implementation redirects them to the logout url and asks them to re-authenticate.

1.4.2 Reoccuring events

Reocurring events were discovered as an issue in the pre-beta user testing. During the design phase when thinking about the calendar, it was forgotten that reocurring events and all-day events could exist.

All-day events do not have the dateTime key response from the Google Calendar API. As a result the code would fail when trying to access the dateTime from the event start date. This resulted in a redesign and re-think of the possible issues which could arise from Google Calendar.

Eventually, the dateTime key was checked and the all day events issue was solved.

However, the reocurring events problem was still existing. When querying for an event, if the event was reocurring then it would group the reocurring events by the first time in which the event was created. This resulted in an image, which was taken on the 12th March 2016 for example, to show events in February - if there was a reocurring event on the 12th March. However, it had an important reoccurence event ID key.

This resulted in a further query being created which would return all the instances that were reocurring. This had to pass in the $event['id']$ to the query to return all these instances; it was filtered down by querying for the start and end date.

When editing a reocurring event, Google calendar performs some unexpected behaviour: instead of silently modifying the event and returning the grouped event, again, it instead returns both the grouped event and the edited event. A succinct solution for this has not been found and has been a slight issue.

1.4.3 Tesseract Confidence

During a meeting with Dr Hannah Dee, it was suggested that some form of confidence score couldbe outputted to the user to show how well Tesseract identifies the text from the image.

The Tesseract command line does not output the confidence of the characters identified; only the C++ library can output the confidence. Due to time constraints, a wrapper for the C++ Tesseract API could not be implemented - so a third party library was chosen, tessocr[CITE].

Tessocr offered the implementation to access the confidence values for the associated words. The algorithm to execute the identification of the characters was quite simple.

Firstly we get all the text lines; Tesseract deals with the lines as a series of text lines. This is then enumerated over and for each line a corresponding list of confidence words is collected from tehe map_all_words API.

Due to this returning a tuple which is an immutable object in Python, modifications on the list could not be made easily. This resulted in the view file checking the tuple content and calculating whether it is above the threshold; 75 for green, 70 for orange and below 65 for red.

1.4.4 Displaying calendar events

1.4.5 Parsing Exif data

1.4.6 Editing calendar events

When adding meta-data to a note, then the date which the user has entered is parsed into a query against the Google Calendar API which would return all the events for that day which they have entered. It would then check the module code entered against the summary field - this would ensure that the events would be found for that given module code.

This potentially displayed the problem of being able to add the note to the wrong event - if there was more than one event with the same module code that day. As a result a further check was conducted to evaluate the date start time and the date which they entered to ensure they matched.

If an event was matched then the event was added to the description field of a given saved note url. One issue which was discovered is that when adding a note it would just replace all the description with the note url and if they were editing it and it no longer existed then it would remove everything from the description. This is naturally bad, as it would mean that a user's description for an event would be overwritten.

This fixed with appending to the end of the description field, for the google calendar. Additionally, a replace was used instead of replacing with an empty string, this preserved the user's content in their description.

1.5 Review against the requirements

Appendices

Appendix A

Testing Results

This appendix chapter shows the different sections of the application that has been tested and the test outcomes.

1.1 Unit tests

1.1.1 Binarise image

```
tests/test_acceptance_homepage.py::TestAcceptanceHomepage::test_once_authorised_it_displays_users_email_address PASSED
tests/test_acceptance_homepage.py::TestAcceptanceHomepage::test_should_display_the_correct_events_in_calendar PASSED
tests/test_acceptance_homepage.py::TestAcceptanceHomepage::test_signing_in_does_not_show_the_sign_in_button PASSED

============================================================================== 3 passed in 9.30 seconds ===============
```

Figure A.1: Acceptance test being conducted for the homepage, to ensure that the homepage displays the correct content.

1.1.2 Calendar item

1.1.3 DateTimeHelper

1.2 Acceptance tests

The following section displays visual representation of the acceptance tests being executed, and their overall status.

1.2.1 Homepage

```
tests/test_acceptance_homepage.py::TestAcceptanceHomepage::test_once_authorised_it_displays_users_email_address PASSED
tests/test_acceptance_homepage.py::TestAcceptanceHomepage::test_should_display_the_correct_events_in_calendar PASSED
tests/test_acceptance_homepage.py::TestAcceptanceHomepage::test_signing_in_does_not_show_the_sign_in_button PASSED

========================================================================== 3 passed in 9.30 seconds ==========
```

Figure A.2: Acceptance test being conducted for the homepage, to ensure that the homepage displays the correct content.

1.2.2 Add meta-data

```
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_clicking_on_date_field_shows_datepicker PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_clicking_on_time_field_shows_timepicker PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_clicking_suggested_lecturer_from_tesseract_populates_lecture_field PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_clicking_suggested_module_code_from_tesseract_populates_module_code_field PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_clicking_suggested_title_from_tesseract_populates_title_field PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_ensure_the_fields_have_required_key PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_form_does_not_show_exif_data_if_image_is_a_png PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_form_exists PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_form_has_correct_url_action PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_form_has_date_of_lecturer_field PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_form_has_lecturer_name_field PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_form_has_location_field PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_form_has_module_field PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_form_has_title_exists PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_form_shows_exif_data_from_image PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_google_calendar_event_shows_when_exif_data_matches PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_google_calendar_response_without_a_date_time_field_ignores_the_response PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_module_field_label_content PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_module_field_label_exists PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_submit_button_exists PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_tesseract_data_is_coloured_correctly_for_confidence PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_tesseract_data_shows_when_image_is_uploaded PASSED

========================================================================== 22 passed in 115.96 seconds ==========
```

Figure A.3: Acceptance test being performed to ensure that meta-data can be added to the correct note.

1.2.3 Edit meta-data

```
tests/test_acceptance_edit_meta_data.py::TestAcceptanceEditMetaData::test_edit_form_is_displayed_on_the_page PASSED
tests/test_acceptance_edit_meta_data.py::TestAcceptanceEditMetaData::test_edit_form_populates_existing_information_correctly PASSED
tests/test_acceptance_edit_meta_data.py::TestAcceptanceEditMetaData::test_ensure_the_fields_have_required_key PASSED
tests/test_acceptance_edit_meta_data.py::TestAcceptanceEditMetaData::test_when_editing_the_date_it_shows_unable_to_save_to_calendar_if_no_event_was_found PASSED
tests/test_acceptance_edit_meta_data.py::TestAcceptanceEditMetaData::test_when_editing_the_date_updates_event_link_should_be_new_html PASSED

========================================================================== 5 passed in 16.21 seconds ==========
```

Figure A.4: Acceptance test being conducted so that a note's meta-data can be edited successfully.

1.2.4 Search

```
tests/test_acceptance_search.py::TestAcceptanceSearch::test_clicking_view_note_shows_the_note_with_meta_data PASSED
tests/test_acceptance_search.py::TestAcceptanceSearch::test_form_with_search_bar_is_displayed PASSED
tests/test_acceptance_search.py::TestAcceptanceSearch::test_notes_not_included_from_other_modules PASSED
tests/test_acceptance_search.py::TestAcceptanceSearch::test_only_display_the_logged_in_users_notes_not_others PASSED
tests/test_acceptance_search.py::TestAcceptanceSearch::test_searching_for_a_module_that_doesnt_exist_return_message PASSED
tests/test_acceptance_search.py::TestAcceptanceSearch::test_searching_for_form_returns_a_note PASSED
tests/test_acceptance_search.py::TestAcceptanceSearch::test_when_searched_for_it_shows_the_user_what_they_have_search PASSED

========================================================================== 7 passed in 29.43 seconds ==========
```

Figure A.5: Acceptance test to ensure that a user can search for a module code and it displays their notes.

1.2.5 Viewing all the notes

```
tests/test_acceptance_view_all_notes.py::TestAcceptanceShowNote::test_to_view_all_notes PASSED

============================================================================== 1 passed in 7.24 seconds ==
```

Figure A.6: Acceptance test being conducted to ensure that all the notes can be viewed.

1.2.6 Show a note

```
tests/test_acceptance_show_note.py::TestAcceptanceShowNote::test_date_values_are_correct PASSED
tests/test_acceptance_show_note.py::TestAcceptanceShowNote::test_delete_link_is_available PASSED
tests/test_acceptance_show_note.py::TestAcceptanceShowNote::test_displaying_whether_event_was_added_a_users_calendar_return_true PASSED
tests/test_acceptance_show_note.py::TestAcceptanceShowNote::test_edit_link_is_available PASSED
tests/test_acceptance_show_note.py::TestAcceptanceShowNote::test_image_loads_on_show_note_page PASSED
tests/test_acceptance_show_note.py::TestAcceptanceShowNote::test_lecturer_name_is_correct PASSED
tests/test_acceptance_show_note.py::TestAcceptanceShowNote::test_location_name_is_correct PASSED
tests/test_acceptance_show_note.py::TestAcceptanceShowNote::test_module_code_is_correct PASSED
tests/test_acceptance_show_note.py::TestAcceptanceShowNote::test_title_value_are_correct PASSED

============================================================================== 9 passed in 42.97 seconds ==============================
```

Figure A.7: Acceptance test being conducted to make sure that a singular note can be viewed correctly.

1.3 Integration tests

1.3.1 Add and edit meta data

```
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_add_meta_data_route_get_request_not_allowed PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_add_meta_data_route_returns_302 PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_add_module_code_via_post_request_successfully PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_edit_route_upload_erroneous_date_format_returns_error PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_edit_route_upload_erroneous_time_format_returns_error PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_get_edit_note_information_returns_200_success PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_it_saves_a_note_object_once_the_meta_data_added PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_once_a_note_is_saved_it_redirects_to_show_note PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_post_to_edit_note_changes_the_foreign_key_association PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_post_to_edit_note_different_data_created_new_meta_data PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_post_with_already_existing_meta_data_should_return_instance PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_posting_exisiting_module_code_new_meta_data_new_instance PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_posting_redirects_back_to_show_note PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_uploading_empty_data_returns_error PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_uploading_erroneous_date_format_returns_error PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_uploading_erroneous_time_format_returns_error PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_using_the_different_module_code_should_save_new_code PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_using_the_same_module_code_as_before_if_one_exists PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_when_session_doesnt_contain_user_id_redirect_homepage PASSED

============================================================================== 19 passed in 6.17 seconds ==========================================
```

Figure A.8: Integration tests carried on the add and edit meta url to ensure the system worked well together.

1.3.2 Homepage

```
tests/test_integration_homepage.py::TestIntegrationHomePage::test_credentials_not_in_session_return_blank_homepage PASSED
tests/test_integration_homepage.py::TestIntegrationHomePage::test_displays_logout_link_if_logged_in PASSED
tests/test_integration_homepage.py::TestIntegrationHomePage::test_home_route PASSED
tests/test_integration_homepage.py::TestIntegrationHomePage::test_if_not_logged_in_it_doesnt_display_logout PASSED
tests/test_integration_homepage.py::TestIntegrationHomePage::test_sign_in_displays_if_not_authorised PASSED

============================================================================== 5 passed in 0.94 seconds ============================
```

Figure A.9: Integration tests conducted on the homepage to ensure that the routes were accessible.

1.3.3 Logout

```
tests/test_integration_logout.py::TestIntegrationLogout::test_logout_removes_the_credentials_key_from_session PASSED
tests/test_integration_logout.py::TestIntegrationLogout::test_logout_removes_the_user_id_from_session PASSED
tests/test_integration_logout.py::TestIntegrationLogout::test_logout_route_does_not_permit_post_requests PASSED
tests/test_integration_logout.py::TestIntegrationLogout::test_logout_route_returns_a_200_error PASSED

============================== 4 passed in 0.77 seconds ==============================
```

Figure A.10: Integration tests conducted for the logout route ensuring the routes are logged out.

1.3.4 Oauth

```
tests/test_integration_oauth.py::TestIntegrationOAuth::test_call_back_route_returns_a_success_status PASSED

============================== 1 passed in 0.65 seconds ==============================
```

Figure A.11: Integration tests conducted for the oAuth route which interacts with the Google API.

1.3.5 Search

```
tests/test_integration_search.py::TestIntegrationSearch::test_if_user_not_in_session_return_to_homepage PASSED
tests/test_integration_search.py::TestIntegrationSearch::test_search_route__with_code_can_not_permit_post_requests PASSED
tests/test_integration_search.py::TestIntegrationSearch::test_search_route_returns_200_status_code PASSED
tests/test_integration_search.py::TestIntegrationSearch::test_search_route_with_code_returns_200_status_code PASSED
tests/test_integration_search.py::TestIntegrationSearch::test_search_with_post_request_returns_405 PASSED

============================== 5 passed in 0.89 seconds ==============================
```

Figure A.12: Integration tests conducted for the search URL to ensure searching works correctly.

1.3.6 Show note

```
tests/test_integration_show_note.py::TestIntegrationShowNote::test_deleting_a_note_deletes_a_note_from_database PASSED
tests/test_integration_show_note.py::TestIntegrationShowNote::test_deleting_a_note_returns_status_code_200 PASSED
tests/test_integration_show_note.py::TestIntegrationShowNote::test_route_returns_status_code_200 PASSED

============================== 3 passed in 0.86 seconds ==============================
```

Figure A.13: Integration tests implemented to ensure that the note can be displayed properly.

1.4 Upload

```
tests/test_integration_upload.py::TestIntegrationUpload::test_get_upload_route PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_put_upload_route PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_saving_file_attached PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_should_not_allow_post_to_show_image_route PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_should_return_200_error_on_404_page PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_should_return_image PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_should_save_the_correct_tif_file_to_upload PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_show_image_route PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_uploading_file_status PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_uploading_right_file_extension PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_uploading_without_file_attached PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_uploading_wrong_file_extension PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_when_uploaded_file_redirects_to_show_image_route PASSED

============================== 13 passed in 5.77 seconds ==============================
```

Figure A.14: Integration tests implemented to ensure that a user can upload their images to the application.

1.5 User

```
tests/test_integration_user.py::TestIntegrationUser::test_user_route PASSED

============================================================================== 1 passed in 0.77 seconds ======
```

Figure A.15: Integration tests implemented the user route is working correctly and a the user gets added to the database.

1.6 View all notes

```
tests/test_integration_view_all_notes.py::TestIntegrationViewAllNotes::test_redirect_to_homepage_if_user_session_not_set PASSED
tests/test_integration_view_all_notes.py::TestIntegrationViewAllNotes::test_show_all_notes_returns_200_success_code PASSED

============================================================================== 2 passed in 0.72 seconds ==============================
```

Figure A.16: Integration tests to make sure the view all notes url is working and getting the appropriate notes from the database.

1.7 User tests

Appendix B

Tesseract data results

This chapter shows the table outputting the results from the Tesseract training phase.

2.1 Table

Experiment	Characters Identified	Characters Correct	Correct Percentage
1	114	70	61.40
2	252	182	72.22
3	345	280	81.15
4	335	265	79.10
5	288	201	69.79
6	276	206	74.63
7	326	256	78.52
8	400	279	69.75
9	462	364	78.78
10	401	266	66.33
11	366	240	65.57
12	362	273	75.41

Table B.1: A table which shows the statistics from the correctly identified characters during the training process.

Appendix C

Example test data

3.1 Calendar week response mock


```
{
"accessRole": "owner",
"defaultReminders": [
  {
    "method": "email",
    "minutes": 30
  },
  {
    "method": "popup",
    "minutes": 30
  }
],
"etag": "\"1234567891012345\"",
"items": [
  {
    "kind": "calendar#event",
    "etag": "\"1234567891012345\"",
    "id": "ideventcalendaritem1",
    "status": "confirmed",
    "htmlLink": "https://www.google.com/calendar/event?testtest",
    "created": "2014-09-10T14:53:25.000Z",
    "updated": "2014-09-10T14:54:12.748Z",
    "summary": "Test Example",
    "creator": {
      "email": "test@gmail.com",
      "displayName": "Tester",
      "self": true
    },
    "organizer": {
      "email": "test@gmail.com",
      "displayName": "Test",
```

```
        "self": true
      },
      "start": {
        "dateTime": "2016-12-01T01:00:00+01:00"
      },
      "end": {
        "dateTime": "2016-12-01T02:30:00+01:00"
      },
      "transparency": "transparent",
      "visibility": "private",
      "iCalUID": "123456789@google.com",
      "sequence": 0,
      "guestsCanInviteOthers": false,
      "guestsCanSeeOtherGuests": false,
      "reminders": {
        "useDefault": true
      }
    }
  ],
  "kind": "calendar#events",
  "nextSyncToken": "synctokenasbebebe=",
  "summary": "test@gmail.com",
  "timeZone": "Europe/London",
  "updated": "2016-03-16T15:13:26.416Z"
}
```


3.2 Google plus response mock


```
{
  "tagline": "Some Dummy data taglone",
  "verified": "False",
  "circledByCount": 100,
  "objectType": "person",
  "emails": [
    {
      "type": "account",
      "value": "test@gmail.com"
    }
  ],
  "occupation": "A Test Occupation"
}
```

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] Bottle, "Bottle: Python Web Framework Bottle 0.13-dev documentation," http://bottlepy.org/docs/dev/index.html, last checked 22nd April 2016.

The Python framework was used as a case-study of potential frameworks to use for the application. Discussed in the design section, but rejected as a choice.

[4] M. Daly, "Mocking External Apis in Python - Matthew Daly's Blog," http://matthewdaly.co.uk/blog/2016/01/26/mocking-external-apis-in-python/, Jan. 2015, last checked 25th April 2016.

A nice simple blog post explaining why hitting an external API is bad, and there should mocking objects instead.

[5] P. Developers, "PEP 8 – Style Guide for Python Code — Python.org," https://www.python.org/dev/peps/pep-0008/, last checked 23rd April 2016.

The PEP8 standard was used throughout the codebase as an implementation style guide. It is referenced in the evaluation to discuss the design decision that should have been implemented from the start of the project.

[6] Django, "The Web framework for perfectionists with deadlines — Django," https://www.djangoproject.com/, last checked 22nd April 2016.

The Python framework was used as a case study, looking at the different frameworks available. It was rejected for it being too large for the project.

[7] 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.

[8] 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.

[9] Flask, "Welcome — Flask (A Python Microframework)," http://flask.pocoo.org/, last checked 22nd April 2016.

The python framework used as an option. Was used in the design section evaluating the decisions that were made. It was used as the choice of framework.

[10] ——, "Testing Flask Applications Flask Documentation (0.10)," http://flask.pocoo.org/docs/0.10/testing/\#accessing-and-modifying-sessions, 2016, last checked 24th April 2016.

The testing documentation for Flask which discusses how session modifications should be handled. Used in the implementation and the testing discussion.

[11] T. P. S. Foundation, "26.5. unittest.mock mock object library," https://docs.python.org/3/library/unittest.mock.html, 2016, last checked 24th April 2016.

The mocking library used throughout the application. Although the documentation is for python 3, it works for python 2.7

[12] M. Fowler, "Mocks Aren't Stubs," http://martinfowler.com/articles/mocksArentStubs.html, last checked 25th April 2016.

When deciding whether mocks or stubs were used, Martin Fowler gave a nice concise answer. It turns out all the tests are mocking the behaviour from the external API.

[13] 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.

[14] 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.

[15] C. Heer, "Flask-Testing Flask-Testing 0.3 documentation," http://pythonhosted.org/Flask-Testing/, last checked 25th April 2016.

The documentation page for the testing library Flask-Testing. It was used throughout the project after a refactor realising it offered better support for testing Flask applications.

[16] 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.

[17] 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.

[18] 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 applicatation to test the user interface and what functionality OneNote offered that may be usedful for the application

[19] ——, "Office LensWindows 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.

[20] ——, "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.

[21] MongoDB, "MongoDB for GIANT Ideas — MongoDB," https://www.mongodb.com/, last checked 22nd April 2016.

The Mongo DB tool used as a comparison for relational database systems and NoSQL ones.

[22] B. Muthukadan, "Selenium with Python - Selenium Python Bindings 2 documentation," https://selenium-python.readthedocs.org/, 2014, last checked 24th April 2016.

The selenium library used for the acceptance tests. It gives good documentation on how to access elements and how to get specific values from the text.

[23] 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.

[24] Oracle, "Overview - The Java EE 6 Tutorial," https://docs.oracle.com/javaee/6/tutorial/doc/bnaaw.html, last checked 22nd April 2016.

An article which discusses the use of Java as a web application language. It reaffirms the point raised that it is good for performance.

[25] 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 survery showing that Google calendar was ranked the most used calendar people use. Added to the analysis stage to justfy why Google calendar was chosen instead of other calendars available.

[26] pytest-dev team, "pytest: helps you write better programs," http://pytest.org/latest/, last checked 24th April 2016.

The library was used throughout the development for reference on testing. It was especially useful for mocking test data.

[27] R. Python, "Headless Selenium Testing with Python and PhantomJS - Real Python," https://realpython.com/blog/python/headless-selenium-testing-with-python-and-phantomjs/, Aug. 2014, last checked 24th April 2016.

A demonstration on how to use Selenium with the Python examples. Additionally references the fact what phantomjs is, and it is a headless browser.

[28] 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.

[29] 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

[30] 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 rentention with note taking, compared to digital text

[31] 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.

[32] 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.

[33] O. Tezer, "SQLite vs MySQL vs PostgreSQL: A Comparison Of Relational Database Management Systems — DigitalOcean," https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems, last checked 22nd April 2016.

Used as a comparison between what relational management system should be used. Used in the design section for a comparision between the different systems presented and evaluated.

[34] Tiaga, "Taiga.io," https://taiga.io/, 2016, last checked TODO.

The project management toold 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.

[35] w3Techs, "Usage Statistics and Market Share of JavaScript for Websites, April 2016," http://w3techs.com/technologies/details/pl-js/all/all, last checked 22nd April 2016.

The website shows a graph of how Javascript has increased its market share on recent web applications. Used as part of the design consideration regarding the use of programming language

[36] 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 teaxonomy is. Clearly labelling it as a classification of a problem.

[37] 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.