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

Design

As the application was developed in an iterative manner, over a series of sprints, class diagrams and design diagrams were not created at the very start of the project. Instead adopting an iterative approach to design was preferred. Regardless of this, some important design decisions were decided at the start of the project. The chapter will clearly explain rationale for the decisions and state whether they were the result of an iterative processes or an upfront design.

1.1 Overall architecture

This section discusses the architecture of the web application. The design for the web application was developed over a series of sprints, iteratively increasing with each user story, therefore no upfront design was conducted at the start of the process.

1.1.1 Class Diagram

An overview of the resulting design of the class diagram is presented, with rationale for decisions made and how an iterative approach was used to reach the concluded design. The class diagram can be found in Appendix G, section 7.1.

1.1.1.1 Justification of design

The following section discusses the appropriateness of the design and any justifications needed. Overall, the design clearly shows the object oriented principle of low coupling high cohesion being used on the project.

Google Services
During early iterations, the Google Calendar API was only going to be utilised to parse the user's calendar events. As a result, the class `GoogleCalendarService` was created - this would ensure that the logic encapsulating the Google calendar was centralised into one class. With the version number and API unlikely to change, constants were chosen as the best identifier; if the URLs and version number were to change in the future it would be easy to change these. The initial purpose of the class was to perform key operations to extract the events, as shown with the

function get_events_based_on_date. The functions themselves were iteratively developed, initially only using the execute_request and get_list_of_events. Due to the scope changing with complexity, in the latter sprints further functions were added.

Initially users were not considered a core part of the system. However, the user story to incorporate users into the system was created. Upon creation it was clear that another class to integrate with the Google Plus API would be required. This class followed a similar structure to the calendar class, except for parsing a user's email, so that it can be persisted in the database.

Eventually, the design was evaluated and duplicated functionality amongst the methods was discovered. In both of the classes the build and execute_request functions were duplicated, without having class specific content. As a result, the extract class refactoring technique [20], was used to create a super class BaseGoogleService. This class encapsulates logic for building and executing queries. Extracting these functions to a super class ensured that pure logic for data and query manipulation can be moved to the GoogleCalendarService and GooglePlusService classes.

Helper classes

Helper classes, in the design, are independent classes that help to modularise the system - whilst grouping related functionality into a single class. As the system grew the duplication of code was beginning to become apparent, so helper classes aid in keeping a design simple.

For example the SessionHelper class was developed to initially store credentials after the oAuth log in, discussed in Section **??**, had been completed and had successfully being appended to the session. The class' functions expanded as further duplication of the session handling was developed into the system. This level of abstraction gave a semantic meaning to the interactions with the session.

Most of the helper classes do not interact with the other classes in the system. There is an exception with the GoogleServicesHelper class. In this class majority of the functions are static. This design decision was induced due to the class not interacting with any specific class level attributes. Furthermore, prior to the implementation of editing a calendar event, the code was dispersed throughout the controllers. In an effort to reduce code duplication this helper class was created - but it was quickly decided that it would just be a proxy for calling specific functions in each of the services classes. Although ideally they should be class level functions, it is appropriate to use static functions.

Persistence classes

The relationships between the persistence classes will not be discussed in this section, see Section 1.4. It is worth noting that designing the persistence classes was again an iterative process through the sprints. For example, for majority of the sprints the title attribute in the NoteMetaData class was not added. It wasn't until the a reflection on the content of a note was conducted that the design changed making this field a required attribute.

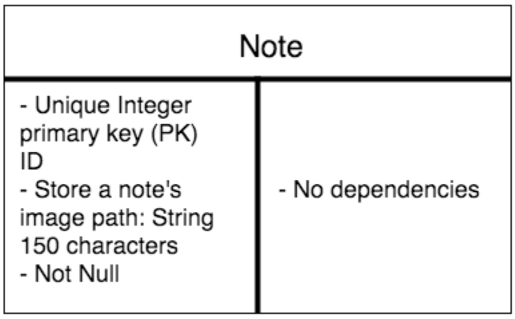
There are a series of save functions in the application, these were added to the design when the controllers were constantly duplicating functionality when persisting object to the database. This improved the readability of the application, providing a succinct solution to persisting an instance to the database.

In parts of the application, information needed to be extracted from the database. To aid in readability, static methods such as find_meta_data were created to keep domain related functionality together, but without creating a specific instance.

Binarisation
The `BinariseImage` class is the model representation of the image segmentation script, as seen in section **??**. The class is called from the controller when a user uploads their image. The output from the class methods is a binarised image. There are a series of functions which integrate with OpenCV API's [30], manipulating an image and performing morphological operations. The class has been constructed so most of the functions are modular.

1.1.2 CRC cards

To aid with the design, class collaboration cards (CRC) were drawn up for each feature. The user-story was decomposed into tasks and each of the tasks had associated CRC cards. This aided in thinking about the design for the class, as well as other classes it interacts with. The overall design discussed in section 1.1.1 is a result of the diligent planning with the CRC cards.



Note	
- Unique Integer primary key (PK) ID - Store a note's image path: String 150 characters - Not Null	- No dependencies

Figure 1.1: An example from Sprint 3, showing a CRC card at the very beginning of creation.

Figure 1.1 shows an example of a CRC card at the very beginning of the note class creation. The left hand side helped to think about methods and attributes for the class. The right hand side shows the responsibilities, where the note may interact with other classes.

Throughout each feature implemented into the system, the CRC cards were created, evaluated and refactored as "throw away designs". Although they were lightweight design tools, they helped to think about the system for the current feature being implemented - reducing the future design creep. For example, during the creation of the note into CRC cards, the image patch attribute was considered to be its own relation. After evaluating that this would be overcomplicating the design for the current feature this design decision was rejected and it would not add benefits to the existing system. The CRC cards were kept for each design and formulated into the class diagram at the end of the development process for formal documentation.

Overall CRC cards were at the forefront of the design during this project. They enabled a clear, concise and well considered design to evolve over a series of sprints. For a further example of an in-depth CRC card see, Appendix H section 8.1.

1.1.3 User interaction

After decomposing the problem that a user would need to be able to add a note, edit and save to a calendar, an activity diagram was constructed to consider the flow of the application.

Throughout the sprints, the design for the activity diagram expanded. The result is depicted in Figure 1.2. The user was not initially part of the application, so the activity did not include the first activity of logging into the system. This was included into diagram once the user story for users must be incorporated into the system was brought forward into the sprint.

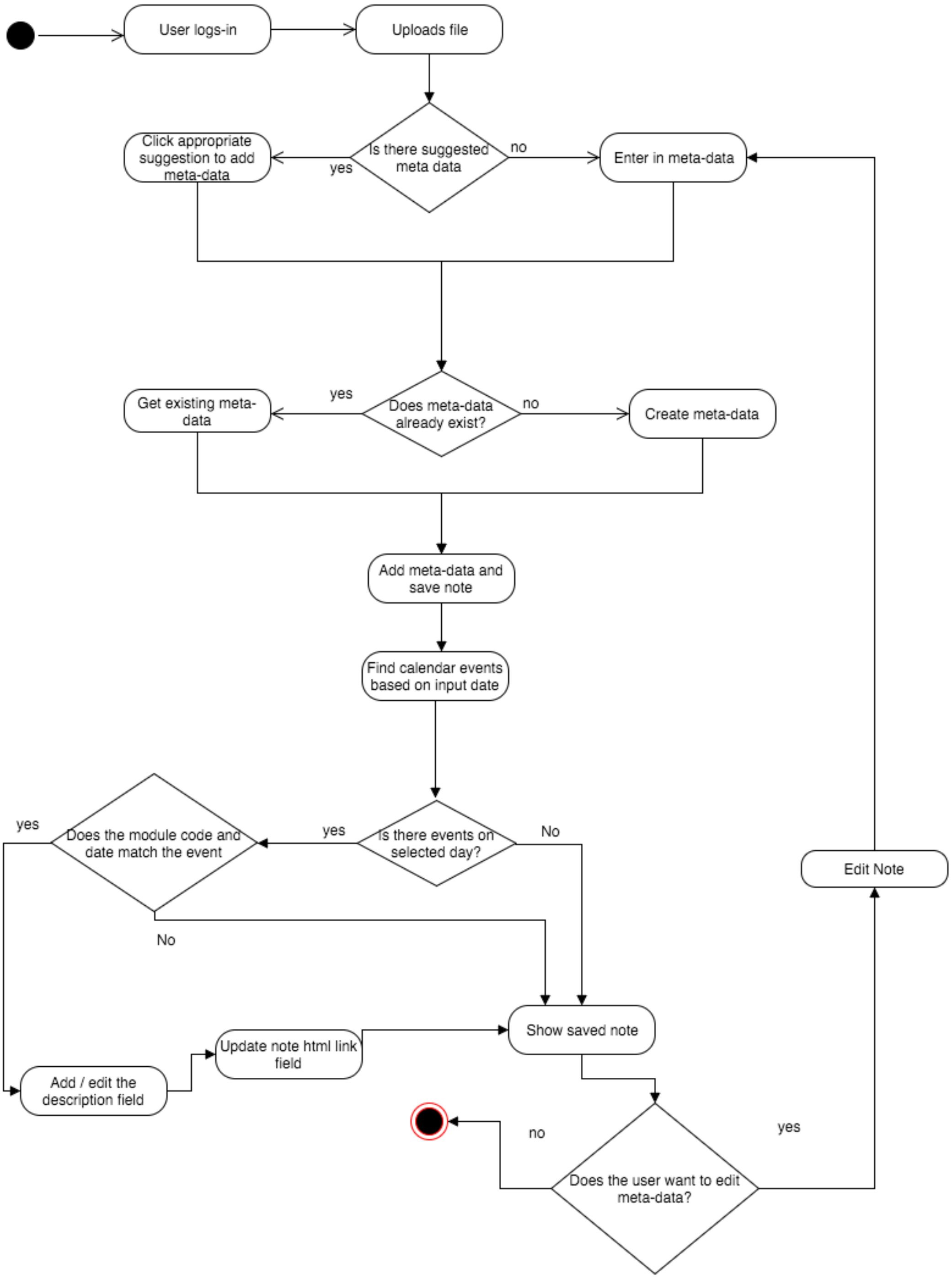

User logs-in
Uploads file
Is there suggested meta data
yes
no
Click appropriate suggestion to add meta-data
Enter in meta-data
Does meta-data already exist?
yes
no
Get existing meta-data
Create meta-data
Add meta-data and save note
Find calendar events based on input date
Is there events on selected day?
yes
No
Does the module code and date match the event
yes
No
Edit Note
Show saved note
Update note html link field
Add / edit the description field
Does the user want to edit meta-data?
no
yes

Figure 1.2: An activity diagram to show how to save a note and the integrations with the calendar.

The conditional checks to identify if there was meta-data outputted from the Tesseract output could not be clicked upon to populate the form. This activity was included into the design during the planning of the feature for that sprint.

Overall, the activity diagram displayed shows the final output of how a note is added into the system. This design has been meticulously developed through a series of iterations to show the final output. It shows that a user can upload an image, they can select any associated meta-data from the suggestions, save the note and it will add it to the calendar item; there is also the option to edit the note.

1.1.4 Model-view-controller

The application would be designed in an Model-View-Controller (MVC) approach. Rationale for different aspects of the MVC structure will be discussed.

1.1.4.1 About MVC

MVC is a design pattern where logic is differentiated from presentation layers, as shown in Figure 1.3.

The controllers aim is not to directly integrate with database and specific logic, instead to interact with a series of models and services. Finally, the controllers will aid in passing dynamic content to view files, returning rendered HTML.

The model in the MVC structure has no acknowledgement of the view file. Instead of rendering any form of HTML, the model is purely data-driven. The sole purpose of the model is to interact with the database and perform any business logic that does not fit in the controller and the view file.

Finally, the view files contain HTML logic with dynamic content passed from the controller. There may be specific logic which impacts the HTML displayed, but no direct calls are made to the database layer or the controller. It uses the dynamic content passed in.

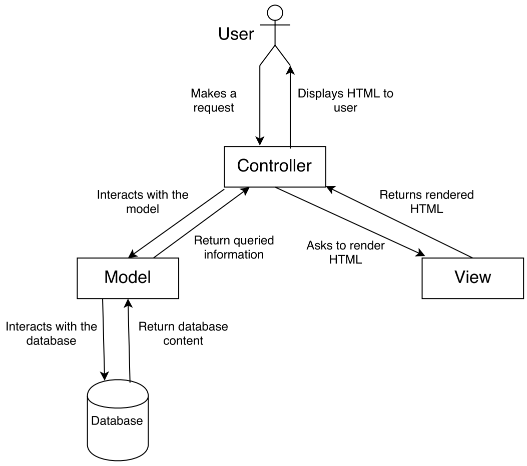


Figure 1.3: A example of how the model-view-controller (MVC) framework integrates.

1.1.4.2 Structuring the web application

Although all the files could not be identified in the design section, the overall structure of the application has been considered.

The primary objective when considering the design of the application would be reusability of the codebase, where applicable. A module based design was considered, where each section of functionality was its own module - but this was rejected as it felt like the codebase would become obfuscated, due to related files - such as views - not being grouped together. Due to this preference an MVC approach would be appropriate - as all the view files can be placed in the same directory.

The framework chosen, see Section 1.6.2, does not support an MVC structure out of the box. Routes are expected to be placed in a singular file; this philosophy is carried through to the models. This was not chosen as the structure of the application as it reduces the clarity of what the code purpose. It also over-complicates the identification of interdependent classes, as it is not explicitly clear from the imports what class is used.

To overcome this, Blueprints [2] were used. Bluerpints are modularised routes allowing different routing options to be placed in different files. Annotations were used to define the blueprint route - each being its own separate controller.

Models were separated into their own directory, and a one class per file policy was adopted to keep the design clean and simple. This ensured that the related file only represented the one class in the system - this would remove any ambiguity when looking at the directory structure.

It is worth considering the view files. The view files were the only section of the web application structure which underwent an iterative process. Initially, the view files would represent the entire DOM tree in a singular file (duplicating headers, scripts etc). This is not the best design decision as there is core HTML which would not change between the different view files, so there was additional duplication that was redundant.

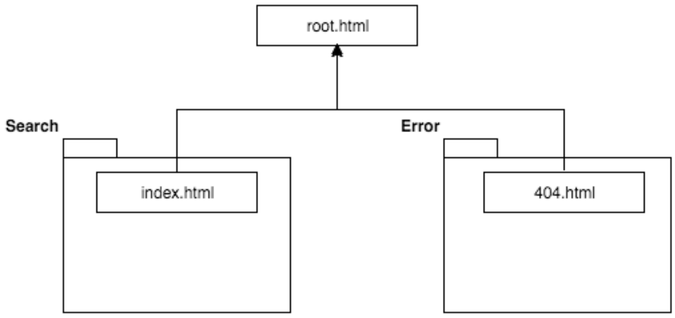


Figure 1.4: A diagram illustrating how extension in Jinga html template engine works.

Figure 1.4 shows the result after the sprint which the design was improved upon. All template files now extend “root.html”, overriding the “content” block. This ensures that the Do not Repeat Yourself (DRY) principle is adhered to and HTML, such as the navigation, are only declared once.

1.1.4.3 Constructing URLs

Often overlooked when considering a design is the URL structure. The design not only aids the developer, but the user interacting with the page can clearly see the intention of that page. Typically there are two types of URLs RESTful-like and query strings.

During the iterations, especially when new functionality was being considered, specific routes were thought about carefully. In the search user-story, query strings were decided to be used. Query strings create URLs such as: `/search?module_code=cs31310`; representing the query string as key-value pairs. During the search feature, it was decided that this approach would be adopted so that the user can easily bookmark the page.

RESTful URLs help to show the a hierarchy of content. Exposing a user to such a URL helps them to clearly identify their content. When the system evolved to displaying a note for a user `/show_note/1`, was chosen for the URL; it is easier to read than `/show_note?note_id=1`. This allows the user to not have additional query parameters to decipher before working out the context of the page.

For the story of viewing notes, it was worth noting that traditional RESTful URLs would be adapted for readability. For example `/view_notes/` was designed, when a proper RESTful URL may be `/notes/`. This offered more semantic meaning to the page's aim.

Overall the design considerations for the URL structure were an important design aspect that was considered to a great deal, to ensure that the user gets the best experience of interacting with the application as possible.

1.2 Image processing

In the very early sprints, the image processing design went through several substantial iterations. Each of the tasks relating to the user story to binarise an image had design implications.

Early work was conducted to investigate how to prepare images for the Tesseract engine. Imagemagick [29] was initially used by converting the image to greyscale - but this yielded poor results. After further design decisions were made to convert the image to monochrome this still returned too much noise. In the following iteration, the processing step would investigate whether the specific thresholding algorithms would be useful.

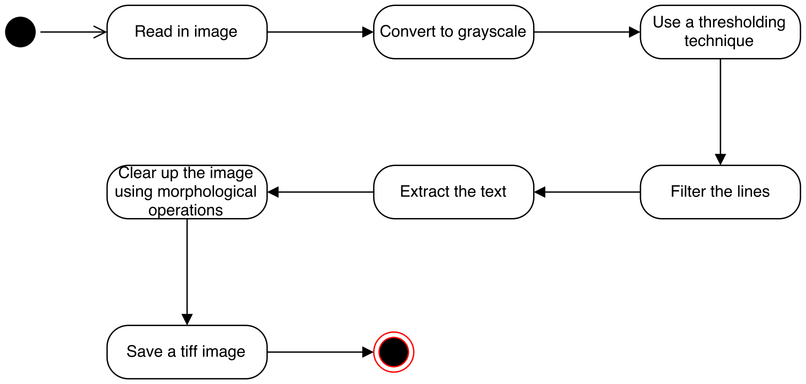


Figure 1.5: An activity diagram to depict the design of the algorithm for the image segmentation.

Figure 1.5 shows the overall activity of how the image processing will be intended to be implemented, after early design work showed binarisation was more complex. Further descriptions of specific implementation can be found in the implementation section **??**.

This high-level activity diagram shows the design stages which were used as a high-level interpretation of the binarisation process. Initially a design was drawn up to just binarise the whole image, but due to implementation issues, this caused too much noise. Therefore, a new algorithm had to be established.

Blue lined paper was one way to overcome this issue. Filtering the lines from a thicker lined paper, would ensure less noise was on the image, creating a better binarised image. Overall, the activity diagram depicts the algorithm of taking a mobile phone photo, filtering the lines, binarising the image and extracting a tiff image. The tiff was selected as a design decision, as Tesseract input requires a tiff file.

This design initially considered blue lines to be important, but it was producing too much noise in the implementation. As a result, instead of trying to extract the lines, it was decided that the lines should be filtered and should only extract the text. This was the final iteration of development on the binarisation script.

1.3 Tesseract

During the analysis phase Tesseract was identified as the OCR tool of choice. Patel et al. [43] performs a case study using Tesseract as the OCR tool to analyse printed text in an image. Patel et al., also discuss the comparison against a proprietary OCR tool, Transym [56].

Patel et al. concludes that Transym only yielded a 47% accuracy on 20 images compared to 70% accuracy using the Tesseract engine.

The first few iterations gave significant insight into how the document might me parsed. Due to the complexities with analysing the whole text on the image, it was limited to the first three lines, parsing the most useful information. It was decided that the first three lines were to be extracted forming the information for the metadata. Although design considerations for parsing the image and looking for key words was considered, it was ultimately rejected due to the complexity. Therefore, a structured approach was adopted. Below is an example of how the meta-data needs to be structured for the notes:

Listing 1.1: An example exert from a valid structured note

```
CS31130: This is a title
Date: 28th April 2016 14:00
By: A Lecturer's name
```

It is worth acknowledging that the test-data used for the Tesseract training had a design element attached to it. When considering what the test data should consist of, there had to be a variety in the data. Pangram's, the "quick brown fox" is the most common example, is a good way to represent text as it contains all alphabetical characters [63]. This would give Tesseract the best possible chance at learning different characters - due to there being an abundance of each letter.

1.4 Entity-relation design

Creating CRC cards enabled considerations to be made about the relations and how they are connected. Through each user-story analysed it was reflected upon and determined if it that would affect the entity-relation design.

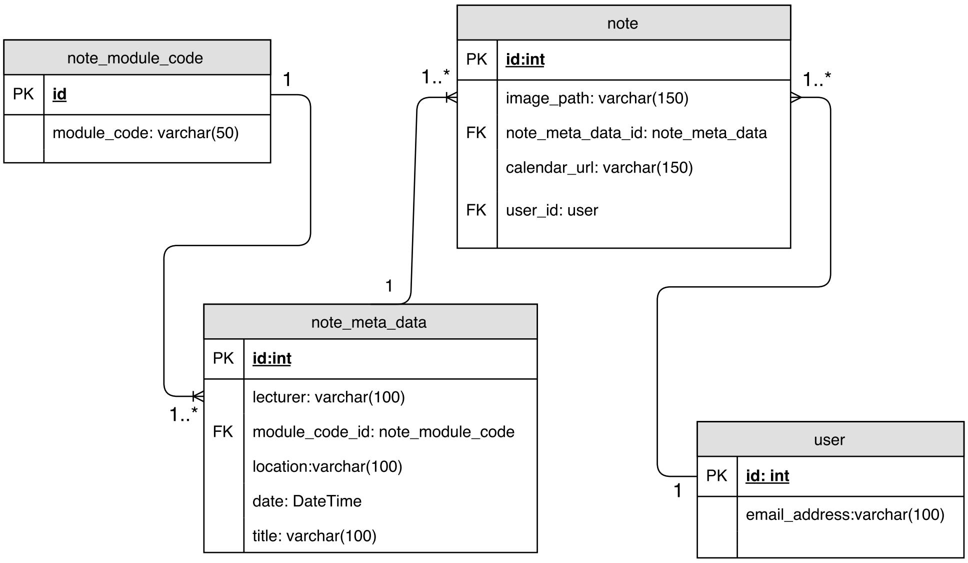


Figure 1.6: The final result of the entity-relation diagram - after a series of iterations.

Figure 1.6, shows the output from the result of final design of the entity-relation diagram. Each new user story added new design implication to the design. For example, when the very basic user-story for creating a note was established, the metadata needed to be added in the future, but the story involved just a note. So no foreign key was created, just the image path attribute.

1.4.1 Justification of design

Below is a justification of the designs through various stories which affected the entity-relation design.

Note

During design persisting the note was one of the first entity-relation design decisions which was made. The attributes selected for the `Note` relation best justify what a note consists of. Firstly, the note contains an image link, which is a relative path to the image. This was persisted to ensure that it could be easily located. When the story for implementing user's was actioned, an additional field containing the user's ID was added to the relation.

During the implementation of adding a URL to a calendar event, the calendar URL was persisted to the database of the associated note. The event ID could have be saved, but the URL was decided to stored so additional queries were not made to the external service. Furthermore, a note will only have one URL.

When implementing the note's metadata, a relation was created and the foreign key was added to the note relation. This was created to ensure that a note must have associated metadata.

Note_meta_data
The `note_meta_data` relation was created in its own relation to reduce data-redundancy, following the principle of normalisation in relational databases. The content could be duplicated for multiple notes, if a user tags the same metadata to more than one note. As denoted from the relationships: a note will have a singular metadata item, but the metadata item could have many notes.

With attributes lecturer, location and datetime - these were the initial design decisions made to be included in the metadata. However, in a later iteration it was decided a title would be preferable; this was added to the relation. The date field is a date-time instead of a string due to integration with the calendar requires specific date-time strings, making it easier to parse.

Initially developed with the module code in this relation, in subsequent iterations the module code was extracted and a foreign key was used.

Module code
The module code was developed into its own relation to prevent data-redundancy. A user may enter multiple notes for the same module code - as a result the database would only need to include one reference of that module code. The relationship between the metadata and the module code is explicit: the metadata must contain one module code but the module code can have more than one metadata item.

User
This was not added to the application until around sprint five. However, the user will have an email address and that would be stored. It is in its own relation due to logic when creating a user: every time a user signs up to the system they are not creating a note instantly, therefore a relation was created to separate this logic. The foreign key was added to a note, so that a note can only have one user - and a user can have multiple notes.

Overall, a succinct collection of relations have been developed which aim to solve the issues of data-redundancy, by providing solid rationale for the resulting design.

1.5 User Interface

With the web application being a core part, a series of User Interface (UI) designs were collated at the start of each breakdown of the story.

The UI had to make the web application feel like an application, rather than a traditional website. This was identified from the background analysis where many systems felt like an application. The colour scheme was aiming to be simplistic, using the Google colour style guide [23]. An alternative of Bootstrap [1] was considered, instead of designing bespoke CSS. Although it has a built-in responsive theme, due to the over-kill of the additional files a simpler approach was adopted.

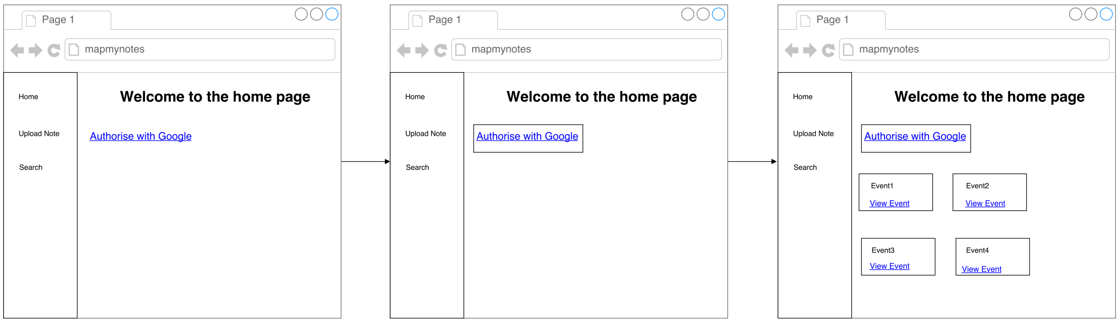


Figure 1.7: From left to right, the homepage wiremock through the different iterations and the change of requirements

Figure 1.7 shows the exploratory wireframe design completed prior to the UI. From the early iterations it was just an authorise button, then a requirement was added to show the user events from the last seven days it was mocked up to reflect this. This process was completed over the stories. If the story reflected a change in the content displayed on the screen a conceptual design was mocked up to ensure there was an idea of how it intended to look.

Further mockups available in Appendix H, Section 8.2.

1.6 Implementation tools

The following sections discuss the implementation tools and their purpose within the application.

1.6.1 Programming language

The programming language would not change per sprint or over an iterative process - as a result this was identified in sprint zero, when additional spike work was completed.

As a web application was being developed investigatory work was completed into the suitability of several server-side languages. Traditionally server-side application languages are: PHP, Ruby, Python, C#, Java and JavaScript, which has increased in popularity [60].

Decomposition of the analysis in the early sprints determined that OpenCV would be utilised on the project. OpenCV's source code is written in C++, however Python and Java bindings are available. Additional research was conducted to see if a reliable wrapper for either PHP or Ruby was available, and after a lot of investigation it was concluded there was not.

C++ is not considered a standard web application development language therefore removing it as a viable option for the web application. Java applications are predominately large commercial applications, using a range of enterprise software - often renowned for their performance abilities [42]. This approach felt too cumbersome for a proposed light-weight application.

By being constrained by design decisions to use OpenCV and a reluctance to use Java, then Python was selected as the most suitable language. Python offers a lightweight and an easy to learn syntax that produces readable code, allowing a object-oriented paradigm to be followed. Additionally, its support for OpenCV is sufficient for the application.

1.6.2 Framework

As Python was being used as the language of choice, this narrowed down the frameworks available. Frameworks are useful for handling more complex features like routing and session handling - leaving the developer to focus on more domain specific issues. Exploratory work was completed in the early sprints to find a suitable tool. The frameworks Django [12], Flask [16] and Bottle [8] were evaluated.

Some frameworks constrain the developers to specific implementations through abstracted classes whereas some offer more flexibility. Whilst evaluating Django, an extensive MVC framework, it was concluded that such a large framework was excessive for this application and it was rejected as a choice for the framework.

Flask and Bottle are classified as “micro-frameworks”, offering a lightweight structure, allowing developers to have more control over the structure. On face value, Flask and Bottle appear to be very similar; they are both lightweight with a similar syntax. After evaluating both of the frameworks it was concluded that Flask has a larger support community compared to Bottle - along with more reliable documentation.

As a result, Flask was chosen as the framework which will be used throughout the application. Spike work was completed into evaluating Flask’s viability for the application quickly showing that it was a suitable tool to use.

1.6.3 Continuous integration tools

Continuous Integration (CI) is normally used in development teams to ensure that all code is checked into the repository. As it was changed for a single person project, so did the point of using it; it was used to ensure every commit passed all tests when pushing to the repository.

After identifying CI would be used in the analysis stage, an appropriate tool would have to be chosen. Jenkins [58] was an initial choice; it is a standalone Java application which a repository can be synced to.

Travis CI [57], is a CI tool in the “cloud” which can be synced to a GitHub repository. Tests can be run during every commit of the application and details regarding if it errors, passes or fails is available.

Although there was not much difference between the two tools, Travis did have the advantage that the web interface could be used rather than a standalone application. A disadvantage of Jenkins would be that for each branch a built script would have to be developed; ideally, the CI tool would be a quick set up and go process, not to be lumbered with further changes. As a result, Travis was chosen as the CI of choice.

1.6.4 Version control

Version control was used on the project to ensure that code was under specific versions. The project was created on a private Git [55] repository on GitHub [22]. Git was chosen for its familiarity and GitHub is a well known place for handling Git based solutions; Travis CI integrated well with GitHub.

It is worth making a mention on the Git flow which was used. As each story was implemented a branch would be created in the form of: `feature/<summary_of_story>`, such as `feature/logout`. All branches were checked out from the development branch - ensuring that all features were from up to date commits. With each feature being developed in its own branch it ensured that any changes made would not affect the overall system. This provided a good platform to develop safely, whilst preserving working code.

Once the code was pushed to GitHub, Travis would automatically build the branch - inside the travis.yml file it would run a series of tests on the application. Once the tests had passed a pull request would be made on the branch into development. If this test successfully passes, and it is safe to merge then it was merged to development.

1.6.5 Development environment

The text editor, Atom [21], was used for the majority of the project. It is a lightweight text editor, which provides suitable syntax highlighting. However later in the project, when refactoring became more cumbersome due to the increase in code base - PyCharm community edition [31] was used as it offered better refactoring functionality.

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.

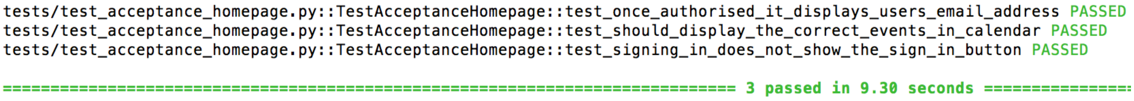
Appendix C

Testing Results

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

3.1 Unit tests

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

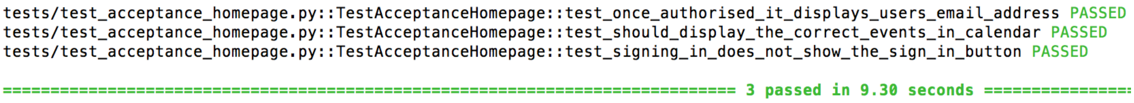
3.1.2 Calendar item

3.1.3 DateTimeHelper

3.2 Acceptance tests

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

3.2.1 Homepage

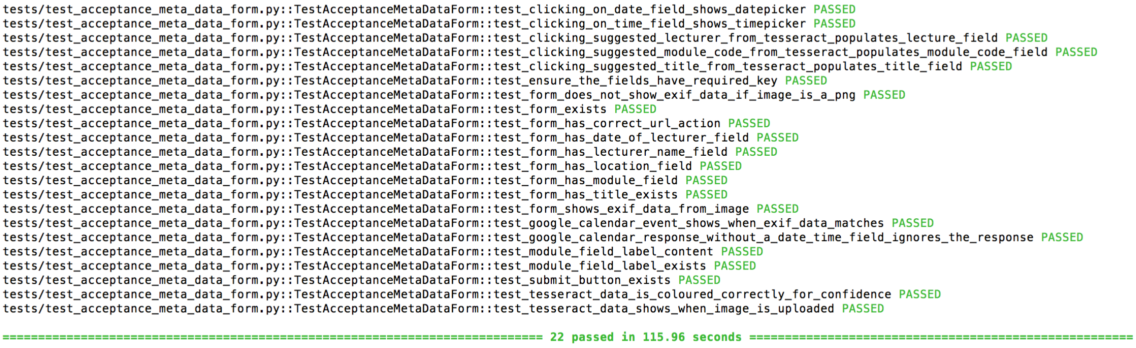


```
tests/test_acceptance_homepage.py::TestAcceptanceHomepage::test_once_authorised_it_displays_users_email_address PASSED
tests/test_acceptance_homepage.py::TestAcceptanceHomepage::test_should_display_the_correct_events_in_calendar PASSED
tests/test_acceptance_homepage.py::TestAcceptanceHomepage::test_signing_in_does_not_show_the_sign_in_button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

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

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

3.2.2 Add meta-data

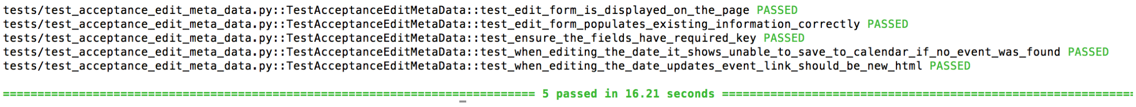


```
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_clicking_on_date_field_shows_datepicker PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_clicking_on_time_field_shows_timepicker PASSED
tests/test_acceptance_meta_data_form.py::TestAcceptanceMetaDataForm::test_clicking_suggested_lecturer_from_tesseract_populates_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 C.3: Acceptance test being performed to ensure that meta-data can be added to the correct note.

3.2.3 Edit meta-data

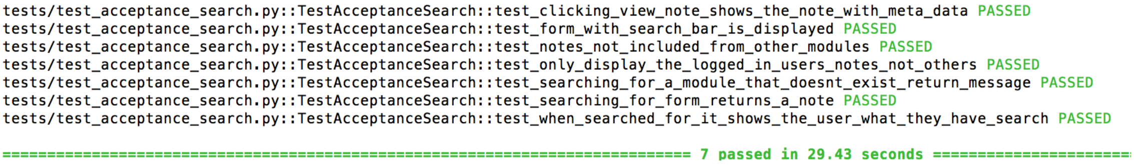


```
tests/test_acceptance_edit_meta_data.py::TestAcceptanceEditMetaData::test_edit_form_is_displayed_on_the_page PASSED
tests/test_acceptance_edit_meta_data.py::TestAcceptanceEditMetaData::test_edit_form_populates_existing_information_correctly PASSED
tests/test_acceptance_edit_meta_data.py::TestAcceptanceEditMetaData::test_ensure_the_fields_have_required_key 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 C.4: Acceptance test being conducted so that a note's meta-data can be edited successfully.

3.2.4 Search

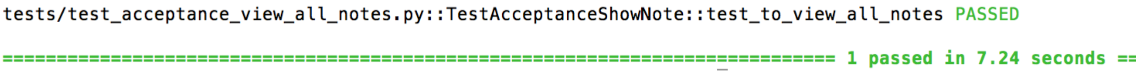


```
tests/test_acceptance_search.py::TestAcceptanceSearch::test_clicking_view_note_shows_the_note_with_meta_data PASSED
tests/test_acceptance_search.py::TestAcceptanceSearch::test_form_with_search_bar_is_displayed PASSED
tests/test_acceptance_search.py::TestAcceptanceSearch::test_notes_not_included_from_other_modules 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 C.5: Acceptance test to ensure that a user can search for a module code and it displays their notes.

3.2.5 Viewing all the notes

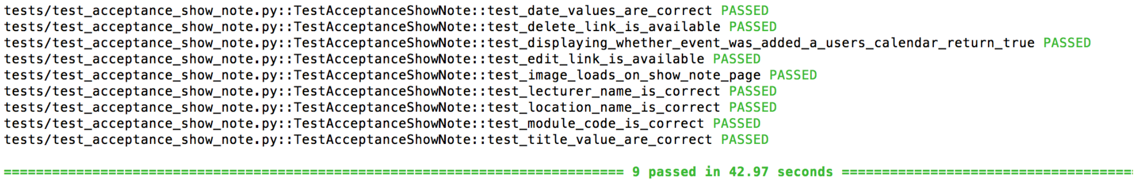


```
tests/test_acceptance_view_all_notes.py::TestAcceptanceShowNote::test_to_view_all_notes PASSED

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

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

3.2.6 Show a note



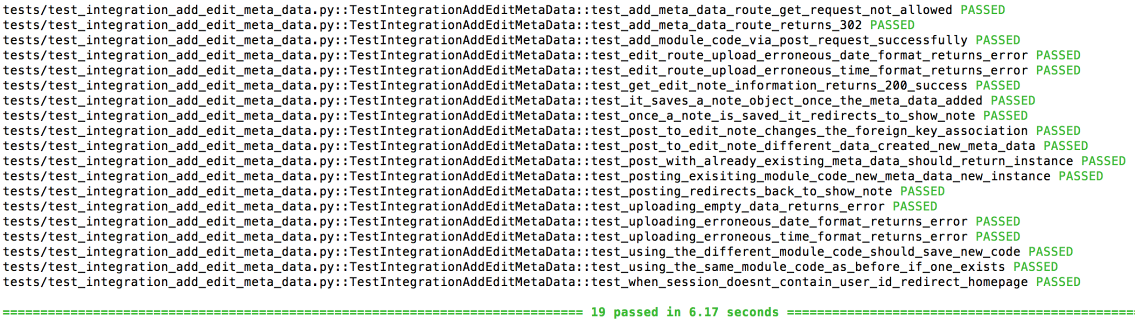
```
tests/test_acceptance_show_note.py::TestAcceptanceShowNote::test_date_values_are_correct PASSED
tests/test_acceptance_show_note.py::TestAcceptanceShowNote::test_delete_link_is_available PASSED
tests/test_acceptance_show_note.py::TestAcceptanceShowNote::test_displaying_whether_event_was_added_a_users_calendar_return_true 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 C.7: Acceptance test being conducted to make sure that a singular note can be viewed correctly.

3.3 Integration tests

3.3.1 Add and edit meta data

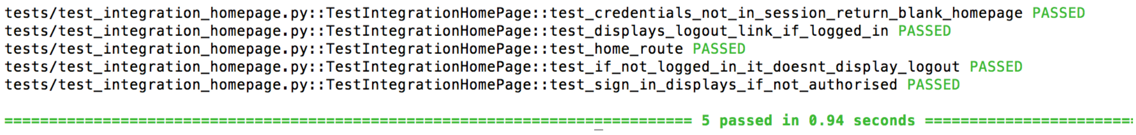


```
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_add_meta_data_route_get_request_not_allowed PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_add_meta_data_route_returns_302 PASSED
tests/test_integration_add_edit_meta_data.py::TestIntegrationAddEditMetaData::test_add_module_code_via_post_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 C.8: Integration tests carried on the add and edit meta url to ensure the system worked well together.

3.3.2 Homepage

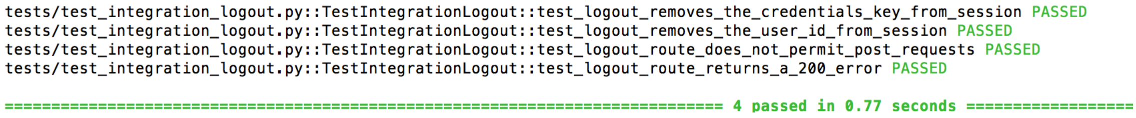


```
tests/test_integration_homepage.py::TestIntegrationHomePage::test_credentials_not_in_session_return_blank_homepage PASSED
tests/test_integration_homepage.py::TestIntegrationHomePage::test_displays_logout_link_if_logged_in PASSED
tests/test_integration_homepage.py::TestIntegrationHomePage::test_home_route 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 C.9: Integration tests conducted on the homepage to ensure that the routes were accessible.

3.3.3 Logout

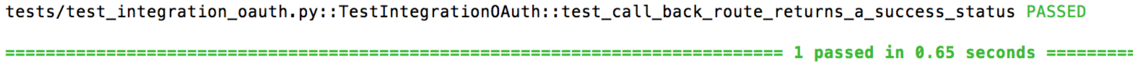


```
tests/test_integration_logout.py::TestIntegrationLogout::test_logout_removes_the_credentials_key_from_session PASSED
tests/test_integration_logout.py::TestIntegrationLogout::test_logout_removes_the_user_id_from_session PASSED
tests/test_integration_logout.py::TestIntegrationLogout::test_logout_route_does_not_permit_post_requests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

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

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

3.3.4 Oauth

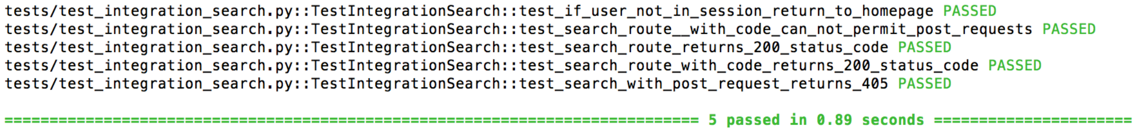


```
tests/test_integration_oauth.py::TestIntegrationOAuth::test_call_back_route_returns_a_success_status PASSED

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

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

3.3.5 Search

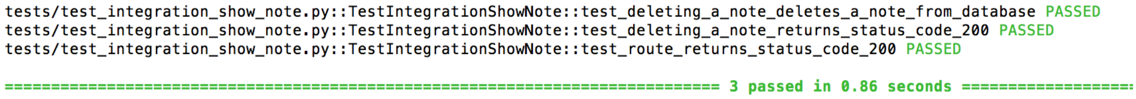


```
tests/test_integration_search.py::TestIntegrationSearch::test_if_user_not_in_session_return_to_homepage PASSED
tests/test_integration_search.py::TestIntegrationSearch::test_search_route__with_code_can_not_permit_post_requests PASSED
tests/test_integration_search.py::TestIntegrationSearch::test_search_route_returns_200_status_code 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 C.12: Integration tests conducted for the search URL to ensure searching works correctly.

3.3.6 Show note

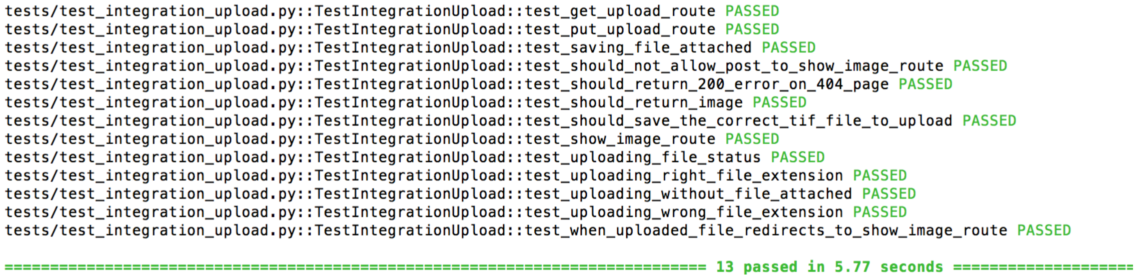


```
tests/test_integration_show_note.py::TestIntegrationShowNote::test_deleting_a_note_deletes_a_note_from_database PASSED
tests/test_integration_show_note.py::TestIntegrationShowNote::test_deleting_a_note_returns_status_code_200 PASSED
tests/test_integration_show_note.py::TestIntegrationShowNote::test_route_returns_status_code_200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

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

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

3.4 Upload

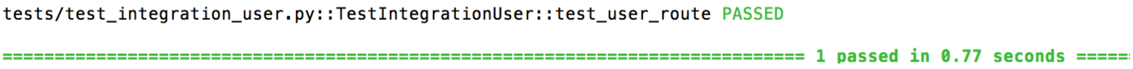


```
tests/test_integration_upload.py::TestIntegrationUpload::test_get_upload_route PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_put_upload_route PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_saving_file_attached PASSED
tests/test_integration_upload.py::TestIntegrationUpload::test_should_not_allow_post_to_show_image_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 C.14: Integration tests implemented to ensure that a user can upload their images to the application.

3.5 User

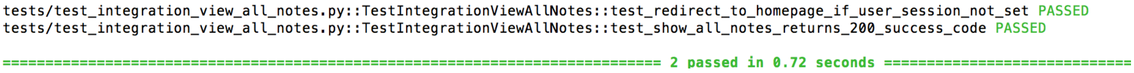


```
tests/test_integration_user.py::TestIntegrationUser::test_user_route PASSED

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

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

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

3.7 User tests

Appendix D

Tesseract data results

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

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

Appendix E

Example test data

5.1 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”
}
```

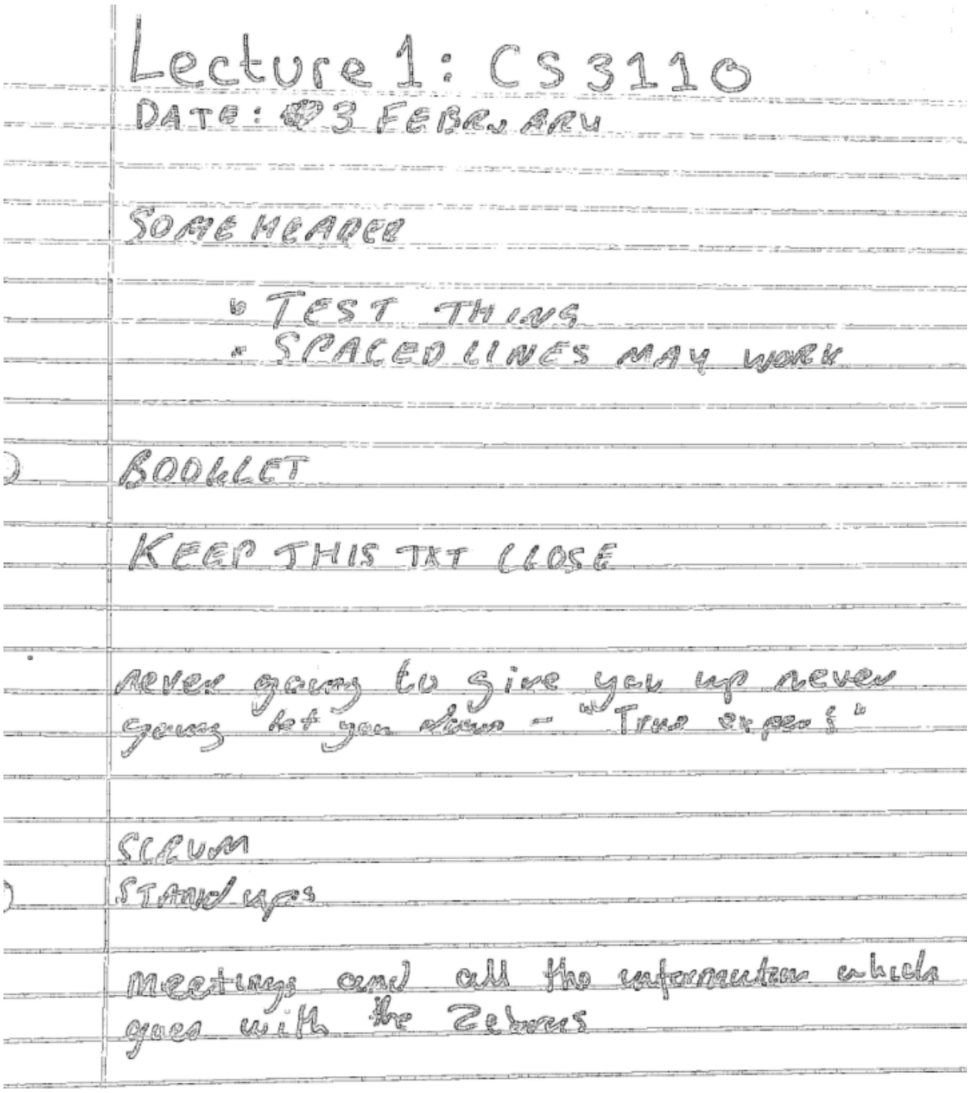
5.2 Google plus response mock

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

Appendix F

Image Processing

6.1 Pre-blue lined image



Lecture 1: CS 3110

DATE: 3 FEBRUARY

SOME HEADER

- TEST THING
- SPACED LINES MAY WORK

BOOKLET

KEEP THIS TXT CLOSE

never going to give you up never going let you down - "True expert"

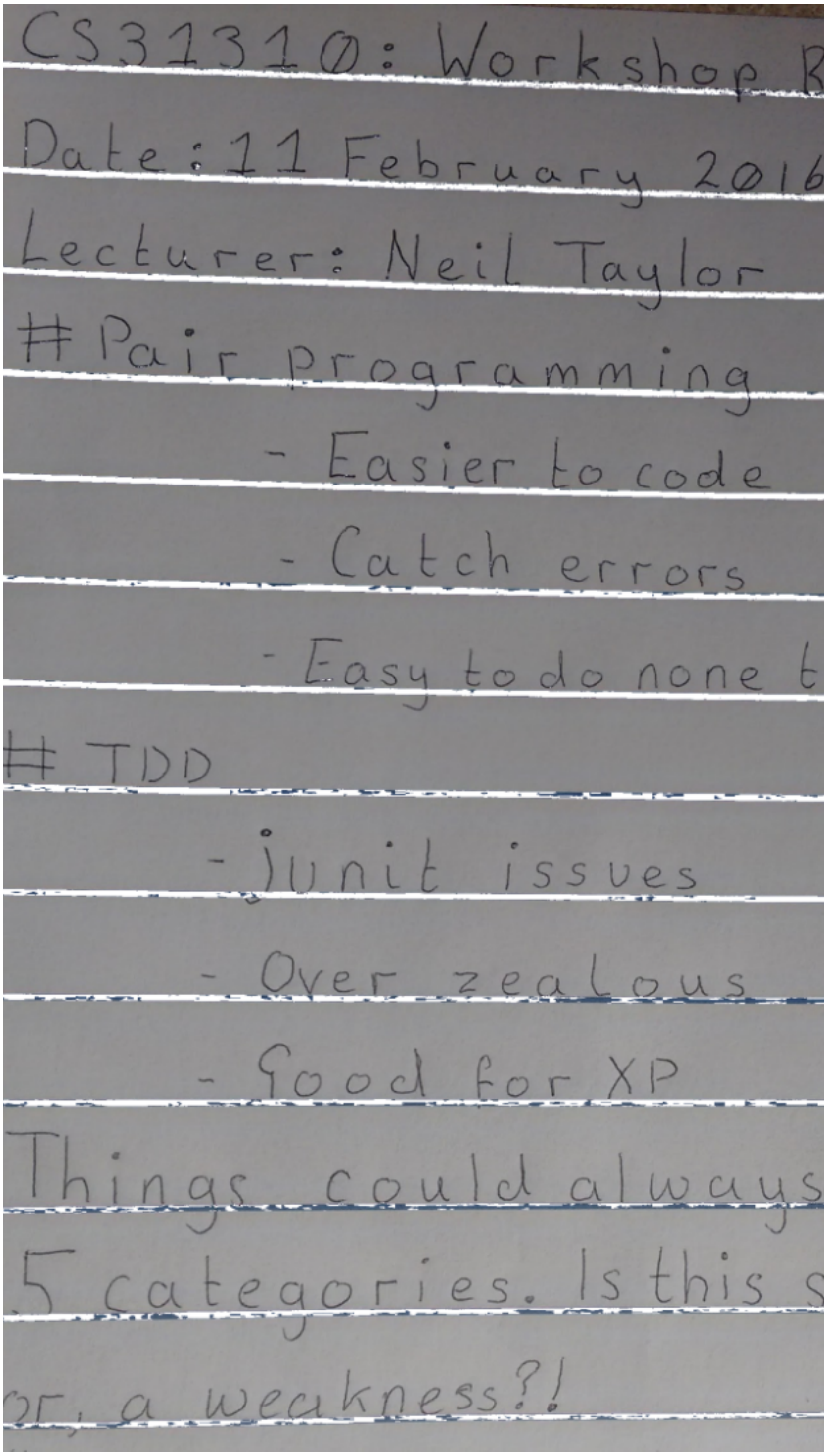
SCRUM

STAND ups

meetings and all the information which goes with the Zebras

Figure F.1: The adaptive threshold on normal lined paper caused too much noise to be interfered with the Tesseract engine

6.2 Filtering the blue lines



CS31310: Workshop R

Date: 11 February 2016

Lecturer: Neil Taylor

#Pair programming

- Easier to code
- Catch errors
- Easy to do none t

#TDD

- junit issues
- Over zealous
- Good for XP

Things could always

5 categories. Is this s

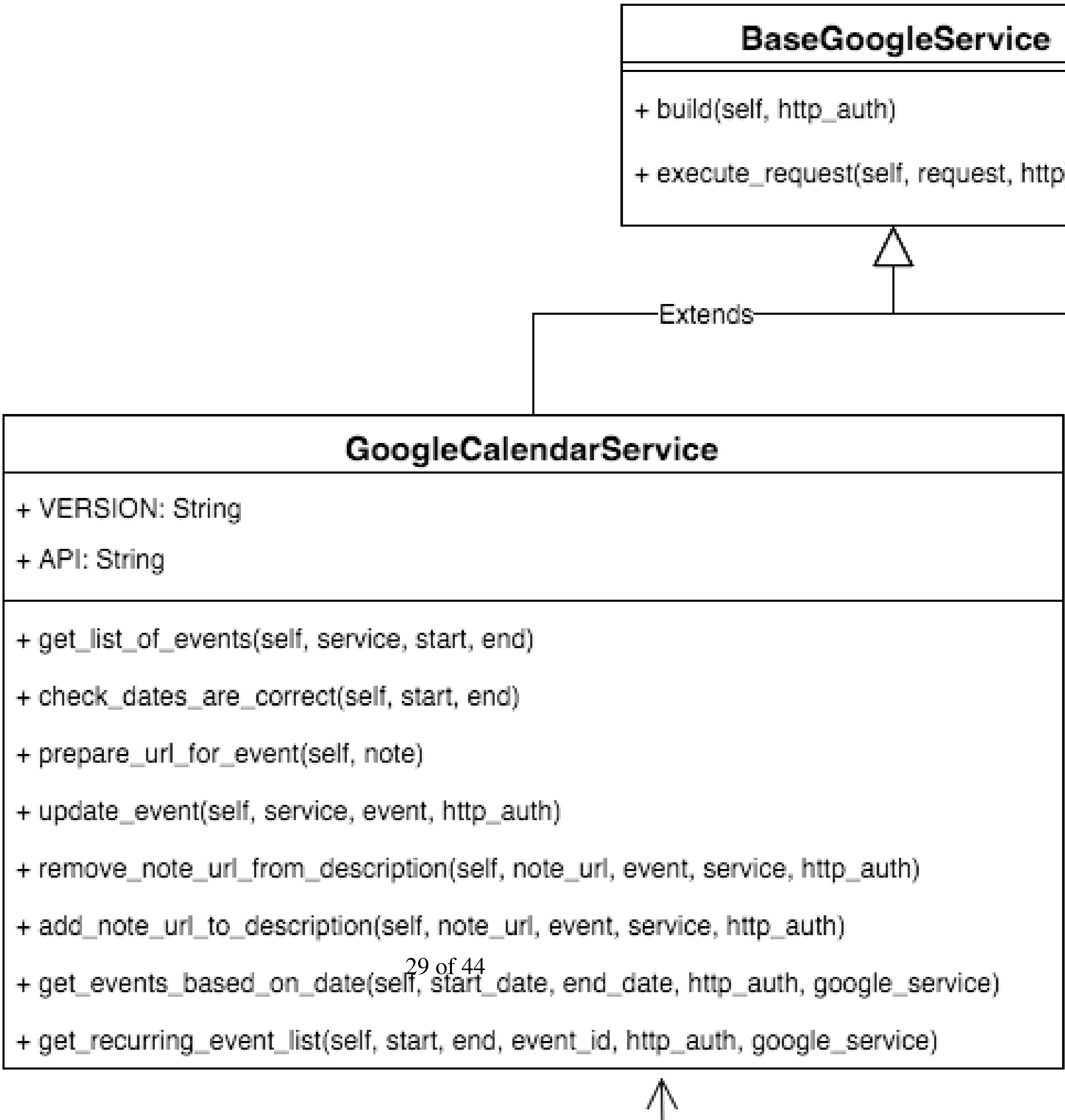
or, a weakness?!

Figure F.2: Blue lines in the adaptive threshold have been identified and removed to be a white colour.

Appendix G

Design decisions

7.1 Class diagram

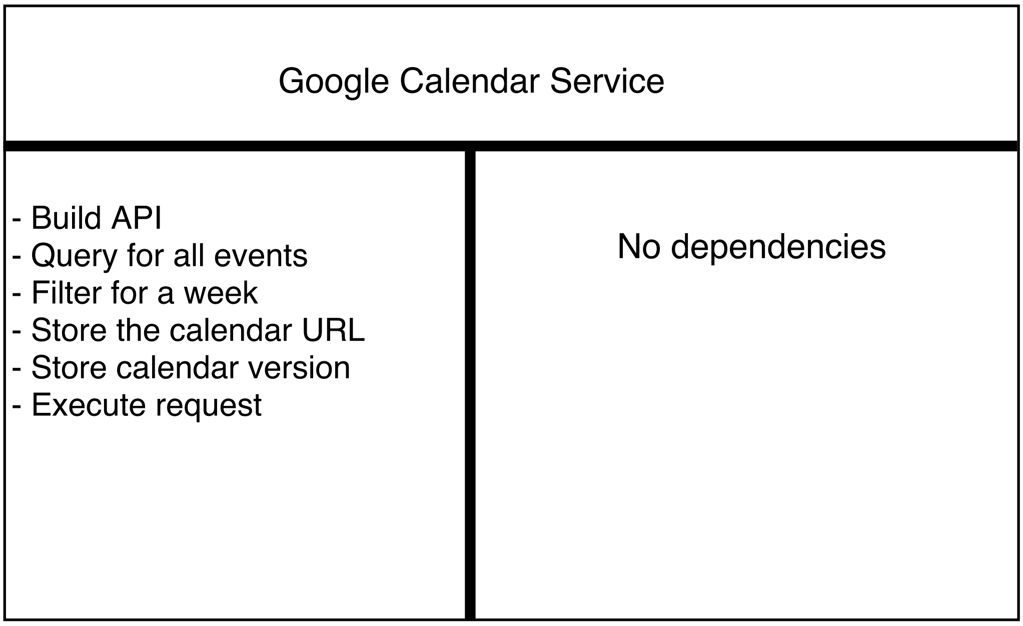

BaseGoogleService
+ build(self, http_auth)
+ execute_request(self, request, http
Extends
GoogleCalendarService
+ VERSION: String
+ API: String
+ get_list_of_events(self, service, start, end)
+ check_dates_are_correct(self, start, end)
+ prepare_url_for_event(self, note)
+ update_event(self, service, event, http_auth)
+ remove_note_url_from_description(self, note_url, event, service, http_auth)
+ add_note_url_to_description(self, note_url, event, service, http_auth)
+ get_events_based_on_date(self, start_date, end_date, http_auth, google_service)
+ get_recurring_event_list(self, start, end, event_id, http_auth, google_service)

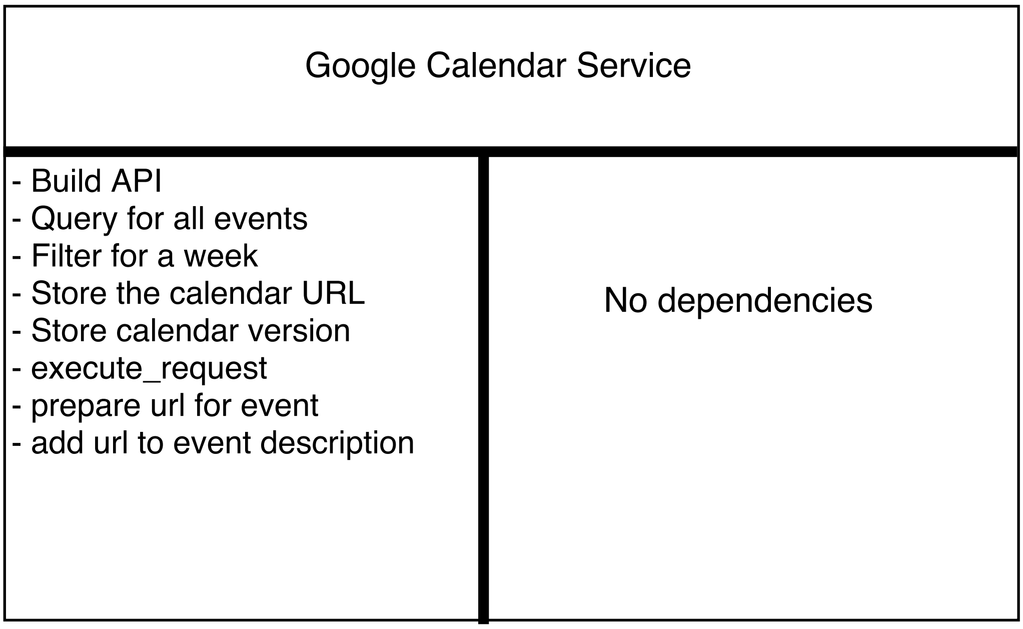
Appendix H

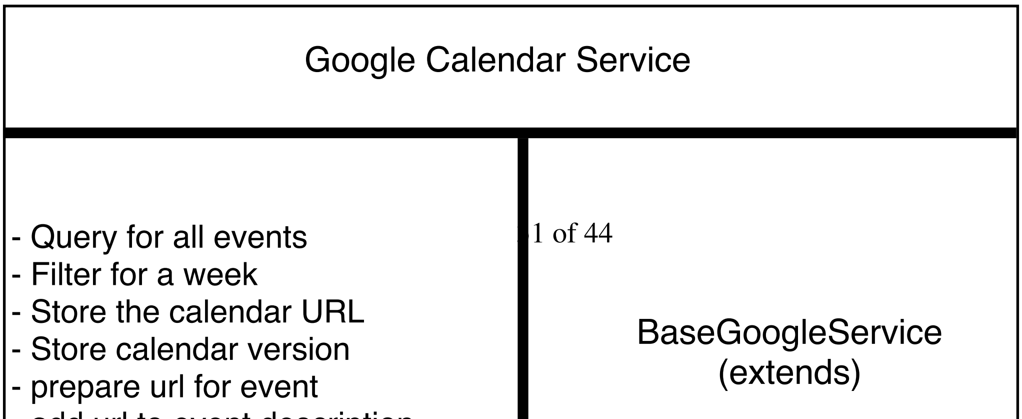
Design suppliments

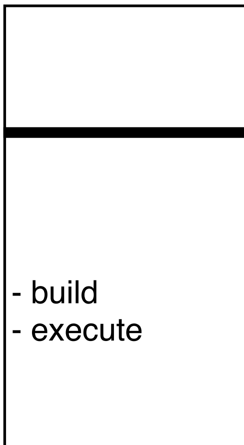
8.1 CRC cards

Below is an excerpt of the the examples of a more complex CRC card design in the system. Throughout the the project, each class went through an iterative process of using CRC cards. Therefore, a lot of them have been omitted to save space.

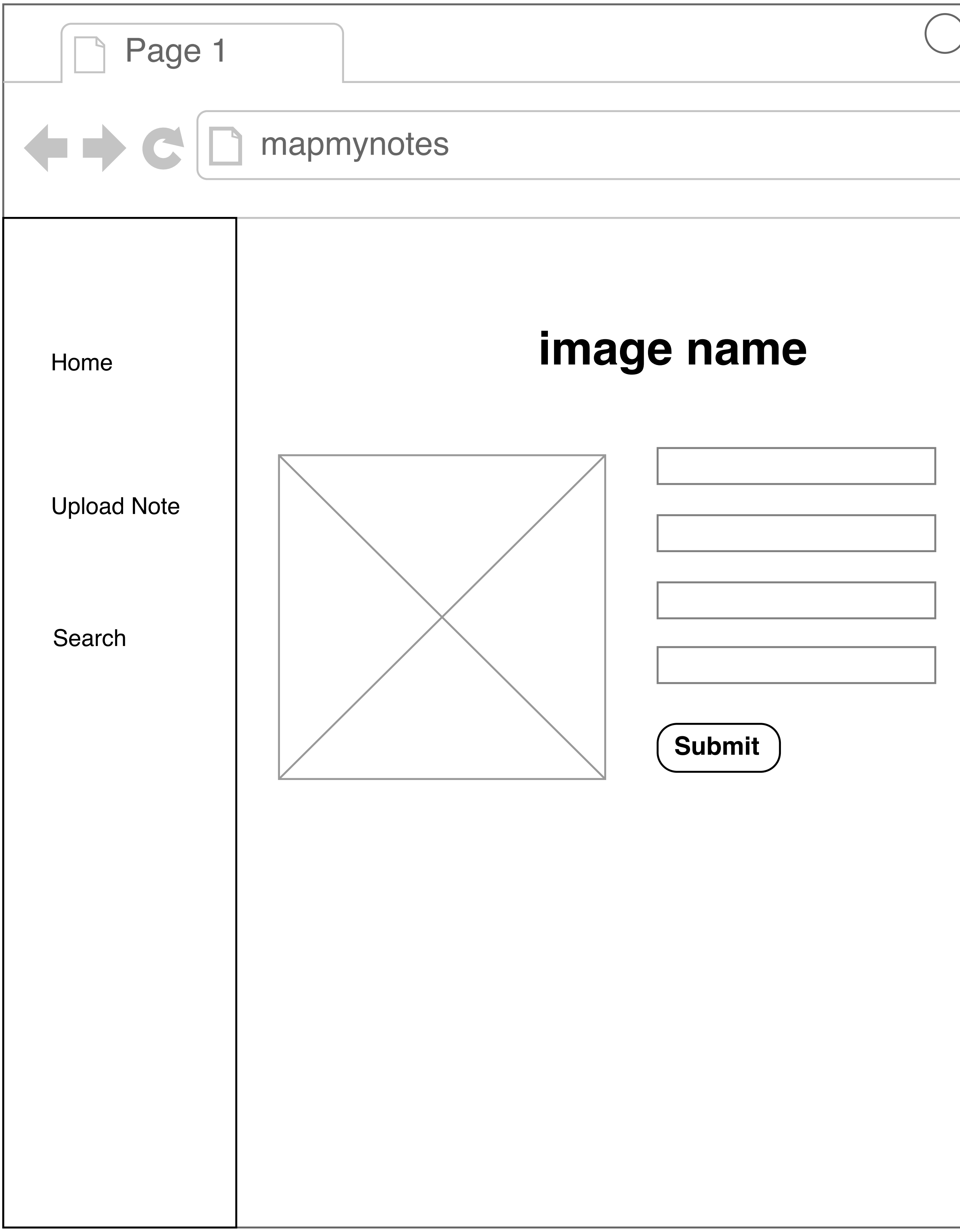

Google Calendar Service
- Build API
- Query for all events
- Filter for a week
- Store the calendar URL
- Store calendar version
- Execute request
No dependencies


Google Calendar Service
- Build API
- Query for all events
- Filter for a week
- Store the calendar URL
- Store calendar version
- execute_request
- prepare url for event
- add url to event description
No dependencies


Google Calendar Service
- Query for all events
- Filter for a week
- Store the calendar URL
- Store calendar version
- prepare url for event
- add url to event description
BaseGoogleService
(extends)


- build
- execute

8.2 Wireframes



Appendix I

Scrum process supplementary materials

The appendix discusses some of the additional material to show the process of scrum used as the methodology of choice. Below is a collection of user-stories throughout the sprints.

9.1 Sprint burndown charts

Figure I.1: An example of the burndown chart for a sprint, showing areas where there may have been difficulty.

9.2 Overall burndown chart

Figure I.2: The overall burndown of the sprints during the development period. This clearly shows a consistent work flow up until more knowledge of the project was achieved, going below the expectation line.

Id	User story	Sprint	Story points
1	As a user I want to be able to upload an image of a set of notes so that I can see my note in the application	2	10
2	As a user I want to be able to tag my notes so that all my notes are under the correct module	4	5
3	As a user I want to be able to add information about the notes so that I can reference them in the future	4	15
4	As a user I want to be able to save a note, so that I can find it again later	3	10
5	As a user I want to be able to search for a given module, so that I can find all notes for that module	7	8
6	As a user I want to be able to sign in via google sign in	5	15
7	As a user I want to use Tesseract OCR so that I can identify characters	1	15
8	As a user I want to be able to view the application on a website	2	5
9	As the customer I want to see the image being binarised properly	2	10
10	As a developer I need to train my handwriting, so that Tesseract can recognise my handwriting	10	n/a
11	As a user I want to be able to edit the meta data, so that I can update it in light of a change	5	5
12	As a user I want to be able to remove a note incase I do not want it to appear	5	5
13	As a developer I want to the website to have good styling	4	8
14	As a developer I want to integrate tesseract into the application, so it can read information from a note	8	8
15	As a user I want to be able to view all the notes I have as a user so I can easily find all of them again	6	3
16	As a user I want to view a list of events on the homepage from my calendar, so I can see recent events	6	15
17	As a user I want to be able to save the URL in the calendars event	7	10
18	As a user I want to be able to tag the title of the lecture, so that I can know which one it is.	6	5
19	As a user, when I authorise I want to show my email address and remove the authorise button, so I know I have signed in	6	3
20	As a developer I want to be able to get the date taken from EXIF data, to show information about a note	7	8
21	As a user I want to be able to edit the date and update my calendar	9	8
22	As a developer I want to be able to associate a note with a user	7	5
23	As a user, I want to be able to have automated suggestion of meta data from the image, so that I can know what to tag.	8	5
24	As a user, I want to be able to logout, so that I can close my session	8	5
25	As a user I want to be able to click Tesseract Items, so that it's easier for me to put in the fields.	9	10
26	As a user I want to be able to edit and save to reoccurring events	10	10

Table I.1: A table showing the user stories identified throughout the project, along with the sprint in which they were implemented and associated story points

Annotated Bibliography

[1] "CSS Bootstrap," last checked 3rd April 2016. [Online]. Available: http://getbootstrap.com/css/

> Bootstrap library was considered when thinking about the styling using CSS.

[2] "Modular Applications with Blueprints Flask Documentation (0.10)," last checked 28th April 2016. [Online]. Available: http://flask.pocoo.org/docs/0.10/blueprints/

> Blueprints were used to modularise the code and expand it for a larger project. They were implemented to attempt to decouple specific routing.

[3] "sirfz/tesserocr: A Python wrapper for the tesseract-ocr API," last checked 25th April 2016. [Online]. Available: https://github.com/sirfz/tesserocr

> The Tesseract wrapper which was used to extract the data from the image. It gives the confidences and all the words associated to the lines.

[4] "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, Jan. 1979. [Online]. Available: http://dx.doi.org/10.1109/tsmc.1979.4310076

> The original paper which OTSU is represented. Although a bit mathematical, some bits were good for reference material on how the algorithm works.

[5] "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

[6] "OpenCV: Extract horizontal and vertical lines by using morphological operations," 2015, last checked 25th April 2016. [Online]. Available: http://docs.opencv.org/3.1.0/d1/dee/tutorial_moprh_lines_detection.html\#gsc.tab=0

> A great reference on how to use different morphological operations and adaptive threshold techniques to extract and binarise an image. Used extensively with the image segmentation script.

[7] 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.

[8] 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.

[9] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.", 2008, pp. 138–139.

A book which explains how the Gaussian function for the adaptive threshold with OpenCV works. It gives a simple description, one which is easy to follow.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] Fisher, "Point Operations - Adaptive Thresholding," 2003, last checked 25th April 2016. [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/HIPR2/adpthrsh.htm

An article explaining clearly and simply how the adaptive thresholding algorithm works. It gives a good level of detail and is concise in its points.

[16] 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.

[17] ——, "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.

[18] 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

[19] 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.

[20] ——, "Extract Class," Oct. 1999, last checked 28th April 2016. [Online]. Available: http://refactoring.com/catalog/extractClass.html

The description of what the extract class refactoring technique, which was used extensively on the project.

[21] GitHub, "Atom," 2016, last checked 28th April 2016. [Online]. Available: https://atom.io/

The text editor which was used for the majority of the project. It lacks refactoring tools, and the application grew too much for a find and search.

[22] ——, "GitHub," 2016, last checked TODO. [Online]. Available: http://www.github.com

The hosting service for the priavte git repository for the application.

[23] Google, "Color - Style - Google design guidelines," 2016, last checked 28th April 2016. [Online]. Available: https://www.google.com/design/spec/style/color.html

The colour guide was used for the CSS colours used throughout the application.

[24] ——, "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.

[25] R. Gouldsmith, “build throws KeyError: ’rootUrl’ error on Google Calendar API Issue #208 google/google-api-python-client,” https://github.com/google/google-api-python-client/issues/208, 2016, last checked 25th April 2016.

A issue which was raised by the author, regarding an issue experienced with a 3rd party library.

[26] ——, “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.

[27] A. Greensted, “Otsu Thresholding - The Lab Book Pages,” http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html, June 2010, last checked 25th April 2016.

A great reference tutorial aiding to identify what OTSU threshold is and how it works in a simple to understand manner, with plenty of example.

[28] 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.

[29] ImageMagick, “ImageMagick: Convert, Edit, Or Compose Bitmap Images,” last checked 28th April 2016. [Online]. Available: http://www.imagemagick.org/script/index.php

ImageMagick is a library which was used for the image binarisation but was not used in the end, due to OpenCV providing better support.

[30] Itseez, “OpenCV — OpenCV,” 2016, last checked 28th April 2016. [Online]. Available: http://opencv.org/

The image processing library used for the image binarisation and the various morphological tools. One of the best tools used on the project.

[31] JetBrains, “PyCharm :: Download Latest Version of PyCharm,” 2016, last checked 28th April 2016. [Online]. Available: https://www.jetbrains.com/pycharm/download/

An IDE used later on in the project to aid in more comprehensive refactoring tools.

[32] 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.

[33] 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.

[34] 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

[35] ——, "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.

[36] ——, "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.

[37] 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.

[38] 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.

[39] H.-F. Ng, "Automatic thresholding for defect detection," *Pattern Recognition Letters*, vol. 27, no. 14, pp. 1644–1649, Oct. 2006, last checked 25th April 2016.

This paper was interesting as it aided in the dicussion of the different thresholding algorithms. It was good to reaffirm some knowledge gained during the development process.

[40] 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.

[41] Opencv, "Miscellaneous Image Transformations OpenCV 2.4.13.0 documentation," 2016, last Checked 25th April 2016. [Online]. Available: http://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous_transformations.html

A description of the adaptive threshold function, which shows that there are two different functionc can be used.

[42] 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.

[43] C. Patel, A. Patel, and D. Patel, "Optical Character Recognition by Open source OCR Tool Tesseract: A Case Study," *International Journal of Computer Applications*, vol. 55, no. 10, pp. 50–56, Oct. 2012, last checked 28th January 2016. [Online]. Available: http://dx.doi.org/10.5120/8794-2784

A great paper on a Tesseract case study tool. It was used as a good comparsion for other OCR technologies as well as providing statistical results for the use of Tesseract.

[44] 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.

[45] 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.

[46] 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.

[47] 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.

[48] 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

[49] 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

[50] 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.

[51] M. Sturgill and S. J. Simske, "An Optical Character Recognition Approach to Qualifying Thresholding Algorithms," in *Proceedings of the Eighth ACM Symposium on Document Engineering*, ser. DocEng '08. New York, NY, USA: ACM, 2008, pp. 263–266. [Online]. Available: http://dx.doi.org/10.1145/1410140.1410197

> A great paper which discusses the Tesseract engine by HP researchers. It is used to discuss the idea that OTSU is used as its pre-processing step.

[52] 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.

[53] 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.

[54] 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.

[55] L. Torvalds, "Git," 2016, last checked 28th April 2016. [Online]. Available: https://git-scm.com/

> The version control management system used on the project, to manage workflows

[56] Transym, "Transym - OCR software for Integrators — Transym Computer Services," 2016, last checked 28th April 2016. [Online]. Available: http://www.transym.com/

> A comparison tool to the Tesseract OCR that is proprietary.

[57] Travis, "Travis CI - Test and Deploy Your Code with Confidence," 2016, last checked 28th April 2016. [Online]. Available: https://travis-ci.org/

> The Travis CI tool which was used during the process. Would be reliably, used and a great tool to aid in the development process.

[58] Various, "Jenkins," 2016, last checked 28th April 2016. [Online]. Available: https://jenkins.io/index.html

> The Jenkins CI tool was considered when analysing which CI tool to use and integrate. Eventually was not chosen because of it being a standalone application.

[59] R. Viet OC, "jTessBoxEditor - Tesseract box editor & trainer," last Accessed 6th February 2016. [Online]. Available: http://vietocr.sourceforge.net/training.html

> An excellent software package which allows the user to train the box files with a great graphical user-interface.

[60] 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

[61] 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.

[62] 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.

[63] F. Words, "Pangrams," last checked 28th April 2016. [Online]. Available: http://www.fun-with-words.com/pangrams.html

> A tool which describes what panagrams are as well as using this tool as inspiration for some section of the training data to ensure that there was a good spread of data.