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

Testing

This chapter discusses the testing strategy which has been implemented on the project. This includes unit, acceptance and user testing utilised throughout various parts of the application.

1.1 Overall Approach to Testing

To recape, an agile approach was adopted throughout the project. From the process, test-driven-development was used throughout the application for almost all aspects of testing.

1.1.1 Test-driven-development

Test-driven-development(TDD) was adopted throughout all aspects of the application. All implementation code as an test which covers the purpose of the implementation. Figure 1.1 shows the TDD cycle.

A sensible test was created and, following the cycle, this would fail when first tested. The following steps would be to ensure that the tests pass by adding the associated code needed to make sure that it passes - afterwards refactoring occurs to ensure that design is kept simple and as clean as possible for the current implementation.

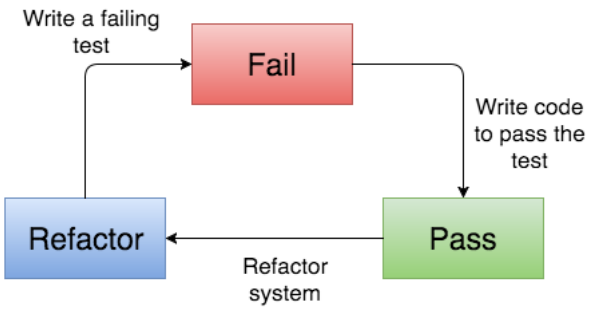

Figure 1.1: The cycle of TDD during the development stages of the application

This approach could have been modified so that a group of tests were created for a feature and then implement a set of functions. This testing strategy was rejected and a pure one test for one bit of functionality was used. This was mainly to ensure that the design was not being over complicated.

Reflecting on the testing strategy and the design, it really did help to think of the design through a test-driven-development way. It ensured that the domain was fully understood before creating a test. This left the codebase tested fully - through a variety of aspects.

1.2 Automated Testing

One thing to note is that Flask's testing documentation is very sparse and is of low quality.

During the first few iterations of tests pytest was originally being used to create test classes, making all the classes extent $unittest.TestCase$.

Flask tests were refactored mid-way through the series of sprints to use Flask-testing[Cite]. This offers better testing support for Flask application, allowing the creation of dummy application and providing the functionality to run a quick server for testing.

1.3 Mocking Tests

Mocking during tests is changing the output of a function to return a specific value which is known about [CITE]. It was established that certain tests would need to be mocked, because some data would change from test to test. It was identified that *all* interactions with Google API's, any interactions with Tesseract and the Session would need to be mocked.

It is best practice when developing application that the developers should not hit a production URL during development and testing. As the Google API does not support specific environment API's, then all URLs would be to a production URL. The issue this raises when testing is ensuring that the tests are isolated and pass every time. For example, if the test queried the API one day it will return a specific result, but when queried another day it may return another result; this requires mocks to be used. The mock would return a specific value every time, ensuring tests can be reliable.

The principle is the same for testing Tesseract in the web application. If more training was conducted then the results from the test on the image would change, therefore ensuring that the tests have consistency data was mocked from specific functions - to check they returned the correct output.

During the first few sprints, whilst understanding how mocks work with Python, there was a lot of duplication with the mocking services. The library mock [CITE] uses annotations above test functions to signal a mock value.

Listing 1.1: An example of using mocks, following the annotation pattern

```
@mock.object(GoogleOauthService, 'authorise')
@mock.object(GoogleCalendarService, 'execute_request')
def test_return_correct_response(self, authorise, calendar_response):
    authorise.return_value = some_json
```

```
calendar_response.return_value = some_more_json
```

Mocking example 1.1 shows the syntax which was initially used by the tests. This would result in many of the tests becomming unreadable and easy to follow exactly what the point was. Additionally the do not repeat yourself (DRY) principle was violated, by duplicating much of the codebase.

Looking the mock API documentation, patching object calls was discovered. Implementing this solution reduced the amount of code for mocking specific functions. The annotations were removed from the top of function tests. Initally it was not entirely clear how to implement these patch functions. Eventually the patch was included in the $setUp$ and $tearDown$ functions, as shown in figure 1.3. The code for mocking is a lot more succinct.

```
def setUp(self):
  # some code
  authorise_patch = mock.patch()
  authorise_mock = authorise_patch.start()
  authorise_mock.return_value = some_json

def teatDown(self):
  mock.patch.stopAll()
```

Often when testing the code there needed to be ways in which the output varied depending on when it was called. In Python mock library they had the functionalty for that, and it was called side effects.

1.3.1 Unit Tests

After refactoring the testing infrastructure to use Flask-testing fraamework instead of the default python unit test, then unit testing became event easier.

The tests would extend the libraries TestCase class. This allows the application access to the application context (the current application) to test models.

Unit tests were conducted for all the classes in the model directory. Model testing themselves was relatively easy and simple to implement.

1.3.2 Route Testing

As the application was using routes then tests were conducted to ensure that the routes could be hit. Whilst doing TDD these were often the first tests to be written. This allowed design considerations to be thought over regarding specific routes.

Other aspects of testing the routes ensured that the correct response codes were returned, ensuring that the routes were working as intended from the very core of the application. Additional tests were added to check that any redirects worked correctly, to ensure flow between controllers were being executes where applicable.

Finally, in some sections persistance testing occured where a test database was checked to make sure that code inside the controllers was being executed. For example, when adding a note the note route checks to make sure that a note was created in the test database.

1.3.3 Handling sessions

One of the tricker aspects regarding testing was the handling of sessions. In parts of the application sessions are used to handle specific states of the system, i.e user logged in.

During testing of routes then session handling would not be too complex. The flask documentation gives a clear breakdown of how to modify a session. To begin with you have to open a test client context - this mirrors what a normal client would look like. From there a session transaction context needs to be opened. Once this has been opened then it would allow for session modification.

Issues arose during the acceptance tests where session modification would need to happen. Issues arose due to the fact that selenium would run on a different process to the application. As a result session modification could not occur during using selenium and acceptance test. This caused a lot of problems regarding testing. As a result, more mocking had to be used. This time the session helper would be then ended up mocking.

Overall, session handling was one of the hardest aspects with testing to get around and find a suitable, yet clean solution.

1.4 Integration Testing

Integration tests would aid in thinking about not just the backend models, but how the information would be presented to the end user.

These tests were implemented in the form of using Python's selenium tool. As previously used before, selenium is a testing tool which allows you to integrate with the DOM and perform automated browser testing. These tests are important to ensure that the user interface is tested for input and ensuring values are correct.

An example of a selenium test to ensure that when the user goes to the file upload page and they're entering information into the meta-data form that associated calendar events are correctlt displayed.

Another useful test where selenium works well is when checking the background colour of the Tesseract output. By mocking the tesseract output the tests were able to test logic such as identifying the confidence score to the correct class name - and therefor the classname being identified with the correct background colour of the text.

1.5 User Testing

Due to the application aiming to solve a problem with a set of students then user studies were conducted and teh responses were analysed and evaluated. Students partook in a usability study which the aim would be to show where the application was good and where it was not so good.

Prior to the actual scheduled user-testing feedback was given regarding the Tesseract output confidence. This "over the shoulder" comments were along the lines of: "It would be great if you could click the identified text and it would automatically populate the text boxes". This was then implemented as a result from pre-user testing.

Further issues which were identified during the user testing were:

- Uploading a JPG image off a phone, which does not have the correct datetime exif key causes the application to fail.
- Uploading an image with a previously uploaded filename caused the application to display the old file.

These issues were caught and modified thanks to extensive user-testing of the application.

One interesting reflection of the user-case study was that people would not use the application. They were quick to defend the applications quality, but the use-case for them taking notes was not present. They much preferred to write up their notes from the lecture for memory retention.

1.6 Tesseract Testing

Due to there being no code written for the Tesseract training process there were no formal tests conducted for this section. However, what could be tested was how well the Tesseract learned as it was going through the learning process.

A test framework was produced which ran the training process on the image. As mentioned, it outputs a text file which is associated to the words identified. This file was then renamed to $eng.ryan.expx_stat.txt$ so that when the final part of the training process is ran it did not overwrite the file.

A statistical analysis was collected on the outputted characters from each of the training examples. Each of the characters were identified on each line of the text file, where each character represents a box which Tesseract has identified as a character.

1.7 Image thresholding Testing

When writing my script for the image thresholding research it was soon discovered that the script would be very trial and error and very much like a protoyping.

When testing the output from the image, then it would be a visual check rather than a specific image that was looking to be achieved. Looking at each itteration through the scripts improvement it would be evaluated as whether it was a worse or better binarisation script. Once a sufficient level of satisfaction from eye-judgement was achieved then the testing from that script was stopped.

Once the spike work for the script had been completed, TDD performed on the image to turn the prototyped script into clean testable code. Using the outputted numpy arrays from open cv, they were compared against what was believed to be outputted.

When the image converts the white image mask to black it performs checks for the image for any black value in the image.

Appendices

Appendix A

Third-Party Code and Libraries

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

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

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

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

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

Appendix B

Ethics Submission

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

Appendix C

Code Examples

3.1 Random Number Generator

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

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

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

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

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

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

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

Annotated Bibliography

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

An article explaining how Evernote does character recognition on images

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

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

[3] 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] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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

[14] ——, "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.

[15] ——, "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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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

[22] 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

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] 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

[28] 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.

[29] 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.