The name of your project

CS39440 Major Project Report

Author: Morgan Jones ([mwj7@aber.ac.uk](mailto:mwj7@aber.ac.uk))

Supervisor: Dr. Chris Loftus (cwl@aber.ac.uk)

22nd April 2019

Version 1.0 (Draft)

This report is submitted as partial fulfilment of a MEng degree in Software Engineering (With Integrated Year In Industry) (G601)

Department of Computer Science

Aberystwyth University

Aberystwyth

Ceredigion

SY23 3DB

Wales, UK

Declaration of originality

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 Registry (AR) 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 Morgan Jones

Date 22/04/2019

Consent to share this work

By including my name below, I hereby agree to this project's report and technical work being made available to other students and academic staff of the Aberystwyth Computer Science Department.

Name Morgan Jones

Date 22/04/2019

Acknowledgements

I am grateful to…

I’d like to thank…

Abstract

Include an abstract for your project. This should be approximately 300 words.

The abstract is an overview of the work you have done. Highlight the purpose of the work and the key outcomes of the work.

Contents

1. Background, Analysis & Process 3

1.1. Background 3

1.2. Analysis 3

1.3. Process 3

2. Design 3

2.1. Overall Architecture 3

2.2. Detailed Design 3

2.2.1. Even More Detail 3

2.3. User Interface Design 3

2.4. Other Relevant Sections 3

3. Implementation 3

4. Testing 3

4.1. Overall Approach to Testing 3

4.2. Automated Testing 3

4.2.1. Unit Tests 3

4.2.2. User Interface Testing 3

4.2.3. Stress Testing 3

4.2.4. Other Types of Testing 3

4.3. Integration Testing 3

4.4. User Testing 3

5. Critical Evaluation 3

6. Annotated Bibliography 3

7. Appendices 3

A. Third-Party Code and Libraries 3

B. Ethics Submission 3

C. Code Samples 3

# Background, Analysis & Process

## Background

The aim of this project is to build a responsive web application that will allow students in the university to give feedback and ask questions anonymously throughout a lecture or workshop. The idea being that the lecturer taking the session could then respond to the feedback and questions as they are provided by perhaps explaining things further or re-visiting misunderstood material. Any data given would be recorded by the system for review by a lecturer at some point in the future enabling the lecturer to easily see a summary of the feedback and adjust future content delivery accordingly.

### Motivation

The project interested me because it presented opportunity for me to learn and gain more experience with web development and in particular the use of a web framework technology, this was something I had struggled with in the past despite having an interest in and knowing that the ability to develop using a full-stack web framework is a valuable, if not essential, skill in the world of web development.

I was also inclined to take the project because of the accessibility of the problem domain. I myself have been a student for years therefore the transition of starting to think like my end users (students) was an easy one.

### Research

During my research I started recording all activity, links and ideas in a project diary that was heavily used during the first few weeks of the project and continued to be used throughout the project.

I knew I wanted to use a web framework approach because my product was to be a web app. The main three I considered was Ruby on Rails [1], Laravel and Django [2] because all three are popular and well established/supported tools. I chose Django in the end despite having previous experience with Ruby on Rails because I am most competent with the python programming language (Django being the web framework for python). This did mean I had to learn Django from scratch, but that learning experience was one of the reasons I chose the project in the first place.

There were many options for my choice of IDE because the standard approach was simply to use a text editor alongside folder/file navigation. The most common editors I saw used in the various Django tutorials I watched were either “Sublime Text 3” or “atom” text editors. I chose “atom” because it has all the functionality of sublime text but with the added benefit of being open source and therefore having many plugins available that I could make use of during the project; I installed a command line console plugin that allowed me to run the server and access the database directly without having to switch desktop windows and a beautify plugin that auto-formatted my HTML and python code.

I knew the system would have to be responsive because most student users would access it through their mobile phones during a lecture. My CSS skills are limited and therefore chose to use a CSS library to assist in this aspect of the project. Bootstrap was my choice of technology for assisting with the responsive design because the purpose of the technology is to “Build responsive, mobile-first projects on the web” [1]. I have also had some experience with older versions of it so had a feel for how it was used already. To revise Bootstrap, I build mock UI designs of how I imagined different pages of the site would look alongside my other research and learning. This allowed me to have a talking point with my supervisors and others about the look of the site it also allowed me to have a detailed starting point for my HTML templates.

The application will need to store data in a database. I had previous experience with MySQL, SQLite3 and PostgreSQL. I chose PostgreSQL because it has the most advanced features and is therefore the most flexible, there is a lot supporting tutorials for using it with Django and it is the database technology I am most competent with having used it all throughout my year in industry.

I wanted to have the feedback display in a visual way and had previous experience using JavaScript along side the HTML canvas to produce client side graphics; it was my aim at the beginning of the project to use these skills to produce some form of visualisation of feedback in graphs or charts.

As inspiration for my system I looked at a quiz system produced by a student (NAME HERE) as a major project in a previous year. It was a good place to start as it showed me the type of style and quality I should be aiming for. The simplistic style of the site and the way he presented data visually stood out to me as something I would want in my app. I also discussed with my supervisor the means of authenticating staff using a login and connection over LDAP to a university server, this was how the student achieved the staff login functionality on his project; a functionality I too would need to implement. As result of this I tried reading the LDAP RFC specifications and investigated the use of a python library (ldap3) that would function as an LDAP API for my Django application. My supervisor also forwarded me an email that was provided to (NAME HERE) with instructions on how to connect and process the data returned by the application level protocol (LDAP).

## Analysis

Taking into account the problem and what you learned from the background work, what was your analysis of the problem? How did your analysis help to decompose the problem into the main tasks that you would undertake? Were there alternative approaches? Why did you choose one approach compared to the alternatives?

There should be a clear statement of the objectives of the work, which you will evaluate at the end of the work.

In most cases, the agreed objectives or requirements will be the result of a compromise between what would ideally have been produced and what was determined to be possible in the time available. A discussion of the process of arriving at the final list is usually appropriate.

As mentioned in the lectures, think about possible security issues for the project topic. Whilst these might not be relevant for all projects, do consider if there are relevant for your project. Where there are relevant security issues, discuss how they will this affect the work that you are doing. Carry forward this discussion into relevant areas for design, implementation and testing.

## Process

I originally intended to follow a personalised version of feature-driven development (FDD) as my engineering process. I chose FDD because it is recommended when building a project which is well defined and would allow me to get a lot of design out of the way during the first three FDD process steps (AKA iteration zero). This approach worked well at the start of my project as it resulted in me producing a range of high-level design diagrams to model the system which served as a useful starting point to which only incremental updates were necessary throughout the project. I also produced an ordered feature list that functioned as a requirements specification and reference point for noting my own progress.

As the project progressed, I found that my progress tracking to be unaligned with FDD because my iterations did not follow the standard 6 FDD iterative milestones I was working through the features on my feature list but was writing code, updating design and tests at the same time. This hinted at my adoption of XP-style iterations and caused me to abandon my attempts at maintaining an FDD progress tracking report.

My iterations were becoming less and less discipled as the project went on as I tried to use different technologies without proper planning, this meant that it became less frequent for me to update tests and design. When viewing the project overall the structure now reflects more of a plan-based approach, with investment in design at the start, heavy investment of most of the time in code/unit test and investment in testing at the end.

I find it hard to maintain

# Design

You should concentrate on the more important aspects of the design. It is essential that an overview is presented before going into detail. As well as describing the design adopted it must also explain what other designs were considered and why they were rejected.

The design should describe what you expected to do and might also explain areas that you had to revise after some investigation.

Typically, for an object-oriented design, the discussion will focus on the choice of objects and classes and the allocation of methods to classes. The use made of reusable components should be described and their source referenced. Particularly important decisions concerning data structures usually affect the architecture of a system and so should be described here.

How much material you include on detailed design and implementation will depend very much on the nature of the project. It should not be padded out. Think about the significant aspects of your system. For example, describe the design of the user interface if it is a critical aspect of your system, or provide detail about methods and data structures that are not trivial. Do not spend time on long lists of trivial items and repetitive descriptions. If in doubt about what is appropriate, speak to your supervisor.

You should also identify any support tools that you used. You should discuss your choice of implementation tools - programming language, compilers, database management system, program development environment, etc.

Some example sub-sections may be as follows, but the specific sections are for you to define.

## Overall Architecture

## Detailed Design

### Even More Detail

## User Interface Design

## Other Relevant Sections

# Implementation

The implementation should discuss any issues you encountered as you tried to implement your design. During the work, you might have found that elements of your design were unnecessary or overly complex; perhaps third-party libraries were available that simplified some of the functions that you intended to implement. If things were easier in some areas, then how did you adapt your project to take account of your findings?

It is more likely that things were more complex than you first thought. In particular, were there any problems or difficulties that you found during implementation that you had to address? Did such problems simply delay you or were they more significant?

You can conclude this section by reviewing the end of the implementation stage against the planned requirements.

# Testing

Detailed descriptions of every test case are definitely not what is required in this section; the place for detailed lists of tests cases is in an appendix. In this section, it is more important to show that you adopted a sensible strategy that was, in principle, capable of testing the system adequately even if you did not have the time to test the system fully.

Provide information in the body of your report and the appendix to explain the testing that has been performed. How does this testing address the requirements and design for the project?

How comprehensive is the testing within the constraints of the project? Are you testing the normal working behaviour? Are you testing the exceptional behaviour, e.g. error conditions? Are you testing security issues if they are relevant for your project?

Have you tested your system on “real users”? For example, if your system is supposed to solve a problem for a business, then it would be appropriate to present your approach to involve the users in the testing process and to record the results that you obtained. Depending on the level of detail, it is likely that you would put any detailed results in an appendix.

Whilst testing with “real users” can be useful, don't see it as a way to shortcut detailed testing of your own. Think about issues discussed in the lectures about until testing, integration testing, etc. User testing without sensible testing of your own is not a useful activity.

The following sections indicate some areas you might include. Other sections may be more appropriate to your project.

## Overall Approach to Testing

## Automated Testing

### Unit Tests

### User Interface Testing

### Stress Testing

### Other Types of Testing

## Integration Testing

## User Testing

# Critical Evaluation

Examiners expect to find a section addressing questions such as:

* Were the requirements correctly identified?
* Were the design decisions correct?
* Could a more suitable set of tools have been chosen?
* How well did the software meet the needs of those who were expecting to use it?
* How well were any other project aims achieved?
* If you were starting again, what would you do differently?

Other questions can be addressed as appropriate for a project.

The questions are an indication of issues you should consider. They are not intended as a specification of a list of sections.

The evaluation is regarded as an important part of the project report; it should demonstrate that you are capable not only of carrying out a piece of work but also of thinking critically about how you did it and how you might have done it better. This is seen as an important part of an honours degree.

There will be good things in the work and aspects of the work that could be improved. As you write this section, identify and discuss the parts of the work that went well and also consider ways in which the work could be improved.

In the latter stages of the module, we will discuss the evaluation. That will probably be around week 9, although that differs each year.

# Annotated Bibliography

|  |  |
| --- | --- |
| [1] | “Bootstrap Documentation Introduction,” Bootstrap, [Online]. Available: https://getbootstrap.com/docs/4.0/getting-started/introduction/. [Accessed 31 January 2019].  *Used to revise the use of the bootstrap 4 CSS library* |

# Appendices

The appendices are for additional content that is useful to support the discussion in the report. It is material that is not necessarily needed in the body of the report, but its inclusion in the appendices makes it easy to access.

For example, if you have developed a Design Specification document as part of a plan-driven approach for the project, then it would be appropriate to include that document as an appendix. In the body of your report you would highlight the most interesting aspects of the design, referring your reader to the full specification for further detail.

If you have taken an agile approach to developing the project, then you may be less likely to have developed a full requirements specification. Perhaps you use stories to keep track of the functionality and the ’future conversations’. It might not be relevant to include all of those in the body of your report. Instead, you might include those in an appendix.

There is a balance to be struck between what is relevant to include in the body of your report and whether additional supporting evidence is appropriate in the appendices. Speak to your supervisor or the module coordinator if you have questions about this.

* 1. 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. If third-party code or libraries are used, your work will build on that to produce notable new work. The key requirement is that we understand what your original work is 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.

The following is an example of what you might say.

**Apache POI library** – The project has been used to read and write Microsoft Excel files (XLS) as part of the interaction with the client’s 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 [5]. The library is released using the Apache License [6]. This library was used without modification.

Include as many declarations as appropriate for your work. The specific wording is less important than the fact that you are declaring the relevant work.

* 1. 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.

* 1. Code Samples

This is an example appendix. Include as many appendices as you need. The appendices do not count towards the overall word count for the report.

For some projects, it might be relevant to include some code extracts in an appendix. You are not expected to put all of your code here - the correct place for all of your code is in the technical submission that is made in addition to the Project Report. However, if there are some notable aspects of the code that you discuss, including that in an appendix might be useful to make it easier for your readers to access.

As a general guide, if you are discussing short extracts of code then you are advised to include such code in the body of the report. If there is a longer extract that is relevant, then you might include it as shown in the following section.

Only include code in the appendix if that code is discussed and referred to in the body of the report.

Random Number Generator

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

// Some example code here…