University of Sussex

Final Report

Final Year Project Database

Author
Lewis Johnson

Supervisor
Dr. Bernhard Reus

2018
BSc Computer Science
Department of Engineering and Informatics

Statement of Originality

....
Signature

Summary

The final year project database is a re-imagined solution to the current website used by the informatics department at the University of Sussex.

In addition to all of the features from the existing website, the new website adds additional helpful features and functionality. Not only were new features added, the website has been rebuilt from the ground up, giving it a fresh and unique look with support for small screen devices such as a mobile phone, which is not present on the existing website.

The application now supports multiple education levels with a single easy log-in, merging the previously separate sites for undergraduate and postgraduate students. No longer is the website restricted for use to the department of informatics, it has been built with everyone in mind. The new website includes support for multiple departments, so any department within the University of Sussex is free to join.

New features for everyone makes the new website easier and more functional than ever before. Newly introduced features include 'Favourite Projects' for students, a personal collection of projects to look back at later; Automatic second supervisor assignment, a feature for project administrators which uses a custom algorithm to pair up students and supervisors; customisation at the administrator level, making it easy to change colours, images and text; and many more features.

Security is also on the up with revamped privileges system. Privileges can be used to restrict certain features and pages to a set of users. Interdependent to privileges is HTTP middleware. Middleware helps prevent attacks by automatically rejecting malicious attacks before they get a chance to reach the database. In addition to said security features, there are form restrictions. Forms may only be submitted by a user who is authorised to do so, and also, every single POST form is automatically sanitised, specially for the MySql database.

Headless browser testing with Laravel Dusk can ensure the website is always in shape, testing navigation, authentication and more. If a test case fails, the issue could be detected and resolved quickly.

Table of Contents

1 Introduction

The purpose of this project is to improve the design, experience, and stability of the informatics final year project database website. The websites main function is to allow final year and master students to pick a project from a curated collection. Currently, undergraduate and postgraduate projects are on separate websites with a duplicate code base.

Sussex faculty, typically lecturers, from the School of Informatics and Engineering can create projects for students to pick, these faculty members are known as ”Supervisors”. Once a student has selected a project, a supervisor can either accept or reject the students’ request. Students also have the ability to propose a project to a supervisor.

The existing solution suffers from multiple problems related to ageing software, also known as bit-rot. For example, there have been occasions where, when a student is accepted for a project, the database has a duplicate entry for the student. The existing solution has some other problems too. Some of these problems include; the website not scaling for mobile devices, bugs on supervisor and administrator side of the website and the ”undo” feature (A feature which lets students and supervisors unselect projects) not functioning correctly. Said issues, and more will be discussed in detail in section 3.2.

The existing solution needs vast enhancements regarding features for all users. Therefore, instead of wasting time trying to adapt the existing solution, it would be better for the solution to be rebuilt from the ground up. Crafting a new solution instead of adapting the existing solution ensures the new website will work on mobile devices and across all modern browsers.

Creating the new solution will require time, and effort to be built in the time-frame given. However, having gained skills from past modules such as ’Software Engineering’, ’Databases’ and ’Web computing’, also having professional experience in web development, the project should be more than manageable. Of course a project this large requires excellent planning and preparation to be delivered on time, and as this is an individual project, self-discipline will be needed to follow the timetable in the project proposal 11.6 faithfully.

In summary, the objective is to implement all features from the existing solution and introduce new features. All of this while making sure the system is completely stable. Please read the requirements analysis 3.3.3 for an in-depth look.

The rest of the report covers the following topics;

- **Professional Considerations** - This section covers the Code of Conduct published by BCS - The Chartered Institute for IT.
- **Requirements Analysis** - This section goes into depth about strengths and weakness of the existing solution. It also covers the client needs, formatted into a formal requirements document.
- **Design** - This section covers all aspects of the design, including front-end, backend and security. It also includes why design decisions were made, and how the design fits with the technical requirements.
- **Development** - The development section is the longest, and one of the most important. This section covers architecture, the implementation of the front-end and backend, accessibility, project management and how the code-base was documented.
- **Security** - The topic of security is scattered throughout the report, nevertheless, this section goes into much more detail about risk management, how attacks will be prevented, input

sanitation and more.

- **Testing** - Any good solution should have great testing, this section covers how Laravel Dusk [30] can be used to provide automated testing with a headless browser, mimicking a real user, and how manual testing for common attacks can be helpful.
- **Deployment** - The section covers how one can deploy this solution to a server for real use. It also covers dependencies, configuration, requirements and more [1].
- **Evaluation** - An evaluation of the finished product.
- **Conclusion** - The conclusion goes over the goals set out for this project and whether or not they were achieved. It also covers extensions which could be added in the future and technologies that may have helped in the development to create a better end product.

2 Professional considerations

In today's world, we are connecting more and more devices to the internet. Smart-phones, smart-homes, smart-drones, smart-domes [40], and all sorts of IoT devices are running software built by engineers from across the globe. Because of rapid globalisation, general morals, and a whole magnitude of other reasons, it has become more crucial than ever that all professional software solutions and research projects are built with ethical standards in mind.

In regards to testing the software, there will be no third parties involved. This means neither any members of the public nor any members of the university will be used to test the software, or gather feedback. The software will be tested exclusively by Dr.Reus and I. Testing the software will make sure all features are stable, and the user interface is accessible for all users. Because no one else is affected, a self-evaluated ethical review seems appropriate for the project.

3 Requirements analysis

This section starts by combing through the existing solution, looking for exceptional features to be kept in the new solution. Additionally there will be a section analysing issues with the existing solution and how they will be circumvented for the new solution. The identified issues may appear to be very minor, but finding every small issue ensures the new solution is the best it can be.

3.1 Terminology

This section provides some clarification on terminology which will be used throughout the document.

3.1.1 Terms

Term	Definition

[1]Found in the appendix

The existing solution	The system currently being used by the School of Engineering and Informatics.
Vanilla *lang*	Vanilla *lang* means the language as how it comes straight from the developers, with no additional frameworks or dependencies.
Hamburger Menu	A type of menu, typically used on mobile apps and websites, indicated with three horizontal lines.
Education Level	Undergraduate, Postgraduate, etc...
Department	This can be any department from any school within the University of Sussex.
Authorised User	*Generally*, a user who is logged in.
User Agent String	A string of text which identifies information about the user, provided by the browsers. This generally contains the device name (e.g. Apple iPhone 8), CPU info (e.g. AMD64), OS info (e.g. Windows NT 10.0), browser info (e.g. Chrome/66.0.3359.117) and compatibility.
UI	User interface.
UX	User experience.
DB	Database.

3.1.2 Technologies

Term	**Definition**
MySql	A relational database management system.
PHP	A Hypertext Preprocessor used as a server-side scripting language for this solution.
Laravel	A 3rd party PHP library, utilised in the new website.
jQuery	A 3rd party JavaScript library, utilised in the new website.
AJAX	Asynchronous JavaScript And XML, a way of sending and receiving data with JavaScript.
CSS	Cascading Style Sheets.
SASS/SCSS	Syntactically Awesome Style Sheets, a CSS preprocessor.
HTML	Hyper Text Markup Language.
DOM	Document Object Model.
ARIA	Accessible Rich Internet Applications.

3.2 Existing solution

The existing solution [33] has been a swift and functional solution for many years and has been used since 2011 by Sussex Informatics faculty and students.

3.2.1 Strength - Speed

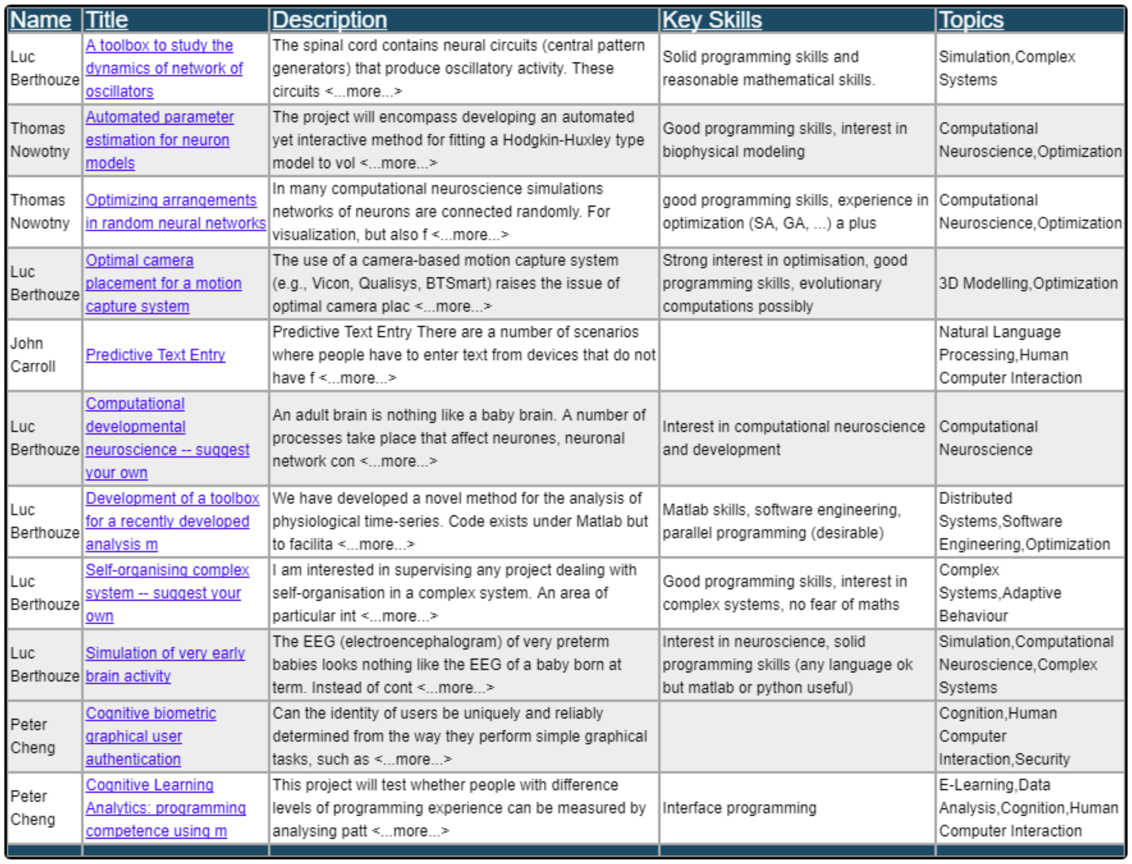
One of the strengths of the existing solution is transfer speed. While the servers are hosted by ITS, the existing solution does a great job of having a small footprint. According to [4], the average website page size in 2015 was 2MB. The existing solutions' total transfer size is a minuscule 43.8 KB, and an even smaller 10.1 KB [2] when the browser has cached files. Users with an incredibly slow network speed could load any page in a near instant. A small footprint is a strength that should be carried over to the new solution.

3.2.2 Strength - Pagination

Another strength of the existing solution is pagination. Pagination is when data is divided into separate documents, so said data can be accessed quicker but in smaller chunks. The existing solution utilises pagination throughout the website for pages which have multiple projects, dividing the projects into a maximum of twenty per page. Because of this, a user can quickly browse through to 100+ projects on the database with ease.

[2]Tested with Chromium network developer tools.

3.2.3 Issue - Displayed Information Is Too Dense



Name	Title	Description	Key Skills	Topics
Luc Berthouze	A toolbox to study the dynamics of network of oscillators	The spinal cord contains neural circuits (central pattern generators) that produce oscillatory activity. These circuits <...more...>	Solid programming skills and reasonable mathematical skills.	Simulation,Complex Systems
Thomas Nowotny	Automated parameter estimation for neuron models	The project will encompass developing an automated yet interactive method for fitting a Hodgkin-Huxley type model to vol <...more...>	Good programming skills, interest in biophysical modeling	Computational Neuroscience,Optimization
Thomas Nowotny	Optimizing arrangements in random neural networks	In many computational neuroscience simulations networks of neurons are connected randomly. For visualization, but also f <...more...>	good programming skills, experience in optimization (SA, GA, ...) a plus	Computational Neuroscience,Optimization
Luc Berthouze	Optimal camera placement for a motion capture system	The use of a camera-based motion capture system (e.g., Vicon, Qualisys, BTSmart) raises the issue of optimal camera plac <...more...>	Strong interest in optimisation, good programming skills, evolutionary computations possibly	3D Modelling,Optimization
John Carroll	Predictive Text Entry	Predictive Text Entry There are a number of scenarios where people have to enter text from devices that do not have f <...more...>		Natural Language Processing,Human Computer Interaction
Luc Berthouze	Computational developmental neuroscience -- suggest your own	An adult brain is nothing like a baby brain. A number of processes take place that affect neurones, neuronal network con <...more...>	Interest in computational neuroscience and development	Computational Neuroscience
Luc Berthouze	Development of a toolbox for a recently developed analysis m	We have developed a novel method for the analysis of physiological time-series. Code exists under Matlab but to facilita <...more...>	Matlab skills, software engineering, parallel programming (desirable)	Distributed Systems,Software Engineering,Optimization
Luc Berthouze	Self-organising complex system -- suggest your own	I am interested in supervising any project dealing with self-organisation in a complex system. An area of particular int <...more...>	Good programming skills, interest in complex systems, no fear of maths	Complex Systems,Adaptive Behaviour
Luc Berthouze	Simulation of very early brain activity	The EEG (electroencephalogram) of very preterm babies looks nothing like the EEG of a baby born at term. Instead of cont <...more...>	Interest in neuroscience, solid programming skills (any language ok but matlab or python useful)	Simulation,Computational Neuroscience,Complex Systems
Peter Cheng	Cognitive biometric graphical user authentication	Can the identity of users be uniquely and reliably determined from the way they perform simple graphical tasks, such as <...more...>		Cognition,Human Computer Interaction,Security
Peter Cheng	Cognitive Learning Analytics: programming competence using m	This project will test whether people with difference levels of programming experience can be measured by analysing patt <...more...>	Interface programming	E-Learning,Data Analysis,Cognition,Human Computer Interaction

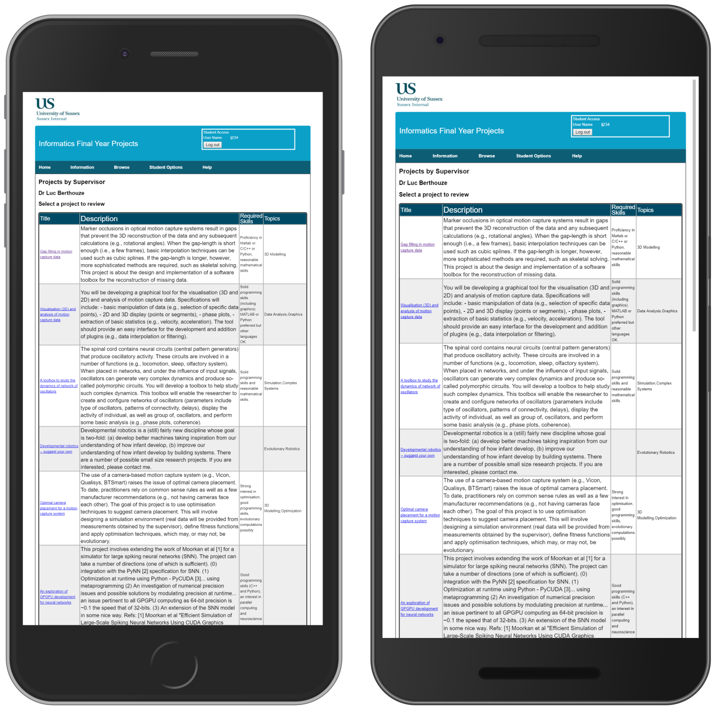
Figure 1: The current "Browse On Offer Projects" page.

Students must look through piles upon piles of projects to find a few they may like, so making this task accessible across all devices is a must. The existing solution displays excess information while browsing projects, and the information could be overwhelming at a glance. The information should be trimmed down, even on desktop devices, which would make it easier for users to browse every project in the database, and even more crucially, enhance the mobile usability and experience.

How was this issue improved upon?

The displayed information was far too dense on this existing solution. In the interim report omitting columns was discussed but a smarter way of approaching this issue arose. Now, users can select which columns they would like to see, with appropriate columns being selected by default. A drop down named 'columns' appears at the top of some tables, where users can select or deselect columns. The primary topic which was discussed before has also been implemented, reducing table density.

3.2.4 Issue - Mobile User Experience



University of Sussex
Sussex Internal

Informatics Final Year Projects

Student Access
User Name lj234
Log out

Home Information Browse Student Options Help

Projects by Supervisor

Dr Luc Berthouze

Select a project to review

Title	Description	Required Skills	Topics
Gap filling in motion capture data	Marker occlusions in optical motion capture systems result in gaps that prevent the 3D reconstruction of the data and any subsequent calculations (e.g., rotational angles). When the gap-length is short enough (i.e., a few frames), basic interpolation techniques can be used such as cubic splines. If the gap-length is longer, however, more sophisticated methods are required, such as skeletal solving. This project is about the design and implementation of a software toolbox for the reconstruction of missing data.	Proficiency in Matlab or C/C++ or Python, reasonable mathematical skills	3D Modelling
Visualisation (3D) and analysis of motion capture data	You will be developing a graphical tool for the visualisation (3D and 2D) and analysis of motion capture data. Specifications will include: - basic manipulation of data (e.g., selection of specific data points), - 2D and 3D display (points or segments), - phase plots, - extraction of basic statistics (e.g., velocity, acceleration). The tool should provide an easy interface for the development and addition of plugins (e.g., data interpolation or filtering).	Solid programming skills (including graphics). MATLAB or Python preferred but other languages OK.	Data Analysis,Graphics
A toolbox to study the dynamics of network of oscillators	The spinal cord contains neural circuits (central pattern generators) that produce oscillatory activity. These circuits are involved in a number of functions (e.g., locomotion, sleep, olfactory system). When placed in networks, and under the influence of input signals, oscillators can generate very complex dynamics and produce so-called polymorphic circuits. You will develop a toolbox to help study such complex dynamics. This toolbox will enable the researcher to create and configure networks of oscillators (parameters include type of oscillators, patterns of connectivity, delays), display the activity of individual, as well as group of, oscillators, and perform some basic analysis (e.g., phase plots, coherence).	Solid programming skills and reasonable mathematical skills.	Simulation,Complex Systems
Developmental robotics -- suggest your own	Developmental robotics is a (still) fairly new discipline whose goal is two-fold: (a) develop better machines taking inspiration from our understanding of how infant develop, (b) improve our understanding of how infant develop by building systems. There are a number of possible small size research projects. If you are interested, please contact me.		Evolutionary Robotics
Optimal camera placement for a motion capture system	The use of a camera-based motion capture system (e.g., Vicon, Qualisys, BTSmart) raises the issue of optimal camera placement. To date, practitioners rely on common sense rules as well as a few manufacturer recommendations (e.g., not having cameras face each other). The goal of this project is to use optimisation techniques to suggest camera placement. This will involve designing a simulation environment (real data will be provided from measurements obtained by the supervisor), define fitness functions and apply optimisation techniques, which may, or may not, be evolutionary.	Strong interest in optimisation, good programming skills, evolutionary computations possibly	3D Modelling,Optimization
An exploration of GPGPU development for neural networks	This project involves extending the work of Moorkan et al [1] for a simulator for large spiking neural networks (SNN). The project can take a number of directions (one of which is sufficient). (0) integration with the PyNN [2] specification for SNN. (1) Optimization at runtime using Python - PyCUDA [3]... using metaprogramming (2) An investigation of numerical precision issues and possible solutions by modulating precision at runtime... an issue pertinent to all GPGPU computing as 64-bit precision is ~0.1 the speed that of 32-bits. (3) An extension of the SNN model in some nice way. Refs: [1] Moorkan et al "Efficient Simulation of Large-Scale Spiking Neural Networks Using CUDA Graphics	Good programming skills (C++ and Python), an interest in parallel computing and neuroscience	

(a) Apple iPhone 6s (b) Google Nexus 6P

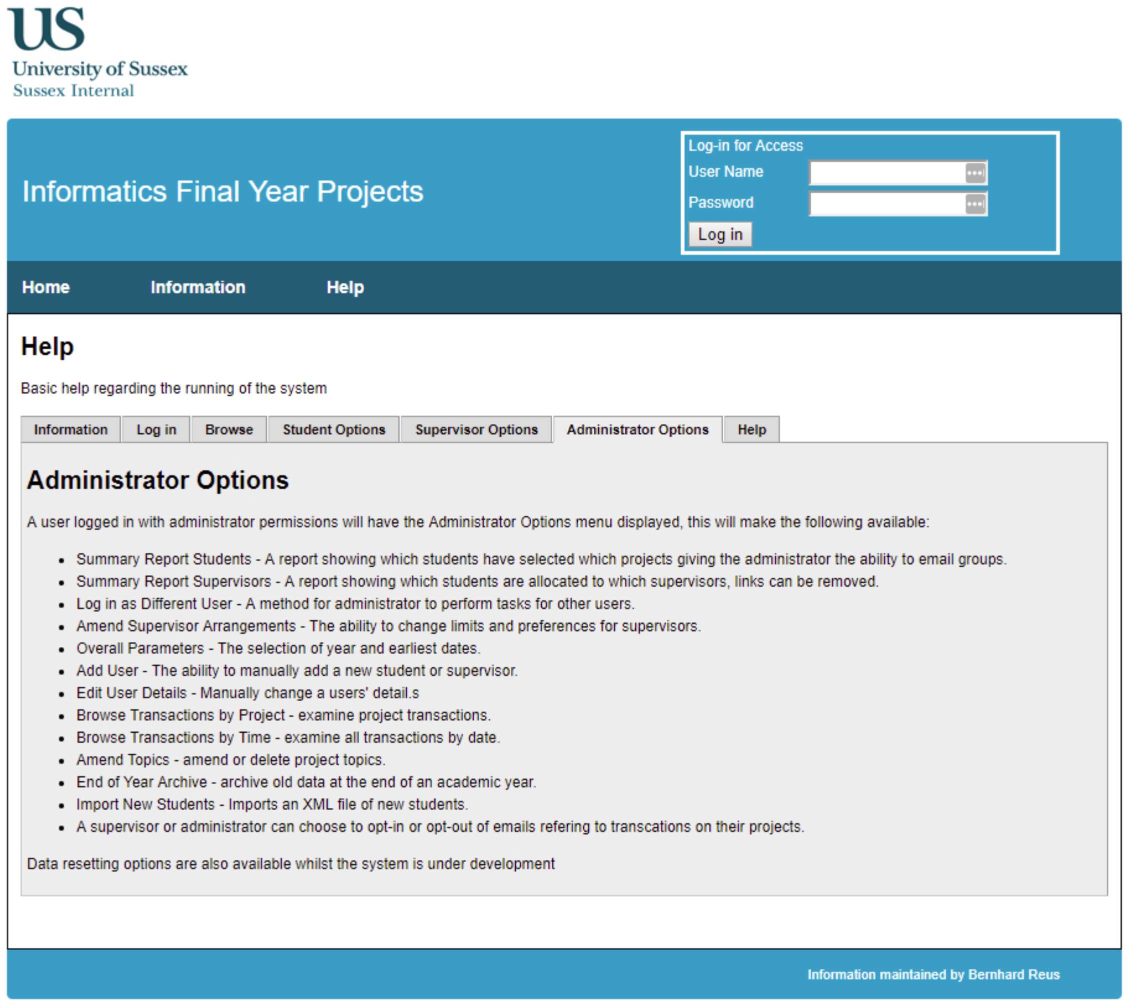
Figure 2: The existing solution on mobile.

As figure 2 demonstrates, the mobile usability and user experience is a bit of a calamity on the existing solution. Because of its age, the existing solution was neither designed nor built with mobile in mind. The site does not scale text to make it readable for mobile, meaning a user will have to zoom in to see the information. Once the page is zoomed in, information in tables will overflow; therefore the user will have to scroll from left to right horizontally. To reassure the statement about the poor mobile experience, see [14].

How was this issue improved upon?

The new solution has been designed for all types of devices from the beginning. There will be multiple user experiences, each tailored for a different type of device, however, the experience will remain consistent. The small screen experience includes a "Hamburger" menu, which is found in many mobile apps. More to come in section 6.4.

3.2.5 Issue - Help User Experience



Help

Basic help regarding the running of the system

Administrator Options

A user logged in with administrator permissions will have the Administrator Options menu displayed, this will make the following available:

- Summary Report Students - A report showing which students have selected which projects giving the administrator the ability to email groups.
- Summary Report Supervisors - A report showing which students are allocated to which supervisors, links can be removed.
- Log in as Different User - A method for administrator to perform tasks for other users.
- Amend Supervisor Arrangements - The ability to change limits and preferences for supervisors.
- Overall Parameters - The selection of year and earliest dates.
- Add User - The ability to manually add a new student or supervisor.
- Edit User Details - Manually change a users' detail.s
- Browse Transactions by Project - examine project transactions.
- Browse Transactions by Time - examine all transactions by date.
- Amend Topics - amend or delete project topics.
- End of Year Archive - archive old data at the end of an academic year.
- Import New Students - Imports an XML file of new students.
- A supervisor or administrator can choose to opt-in or opt-out of emails refering to transcations on their projects.

Data resetting options are also available whilst the system is under development

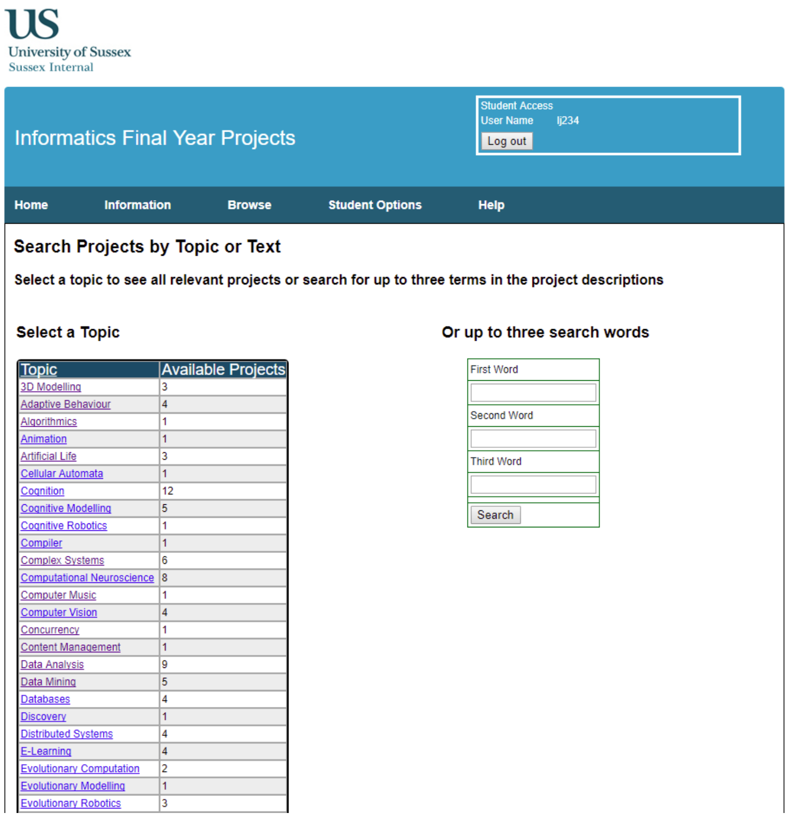
Figure 3: The current "Help" page for an unauthorised user.

At the moment, everyone can see the help sections for all users. This means even guests (Unauthorised users) can see supervisor and administrator help sections. While this is not a huge issue, it could potentially pose a security threat. As described in [38], this can also be considered a poor user experience, because users will notice help sections which are irrelevant to them.

How was this issue improved upon?

The help section now detects which type of user is visiting the page, ensuring only information which is relevant to the user will be displayed. For instance, a student will not be able to see supervisor or administrator help sections.

3.2.6 Issue - Browse by Search Criteria



Informatics Final Year Projects

Search Projects by Topic or Text

Select a topic to see all relevant projects or search for up to three terms in the project descriptions

Select a Topic

Topic	Available Projects
3D Modelling	3
Adaptive Behaviour	4
Algorithmics	1
Animation	1
Artificial Life	3
Cellular Automata	1
Cognition	12
Cognitive Modelling	5
Cognitive Robotics	1
Compiler	1
Complex Systems	6
Computational Neuroscience	8
Computer Music	1
Computer Vision	4
Concurrency	1
Content Management	1
Data Analysis	9
Data Mining	5
Databases	4
Discovery	1
Distributed Systems	4
E-Learning	4
Evolutionary Computation	2
Evolutionary Modelling	1
Evolutionary Robotics	3

Or up to three search words

First Word

Second Word

Third Word

Search

Figure 4: The current ”Browse by Search Criteria” page.

The existing, poorly named, search page, ”Browse by Search Criteria” has two usability issues. The first issue being that a user can browse by topic on this page, this should be on a different page because this is filtering, not searching. The other issue being that the search box is split into three words, limiting how many words a user can search. Additionally, there is no indication of how the query will be used. Because of this, the user will not know if their query is to search for title, supervisor, description or key skills.

How was this issue improved upon?

The first issue was solved by separating the browse by topic function from the search function. The ’browse by topic’ function is now on a separate page, and the search function is on the browse projects page.

The second issue was solved by having a filter and only one search box. The filter has toggles such as Title, Supervisor and Description. Most apps and websites are set up like this, so the experience will hopefully be familiar to most users.

3.2.7 Issue - Authentication

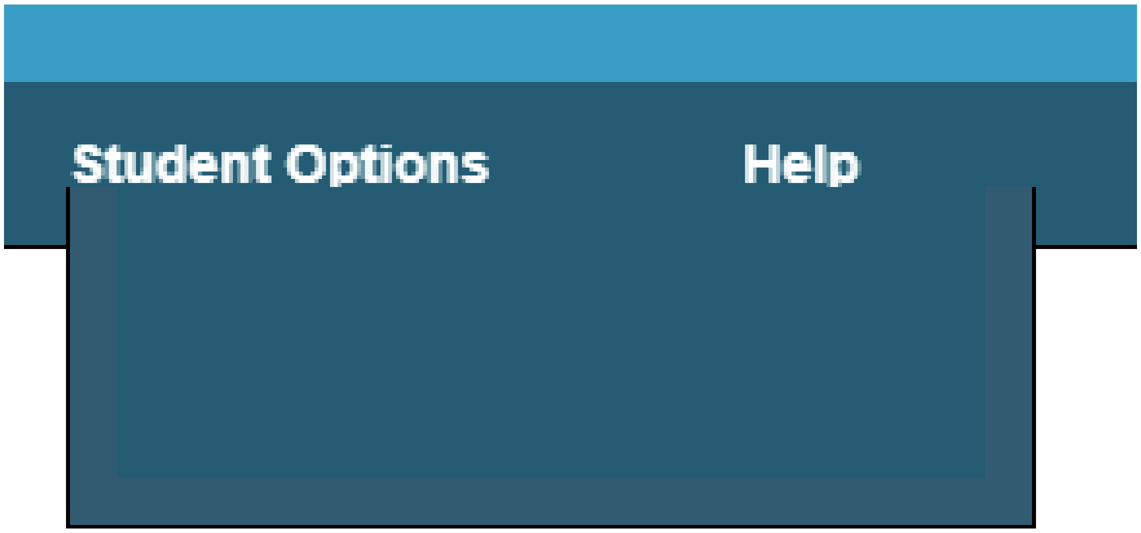


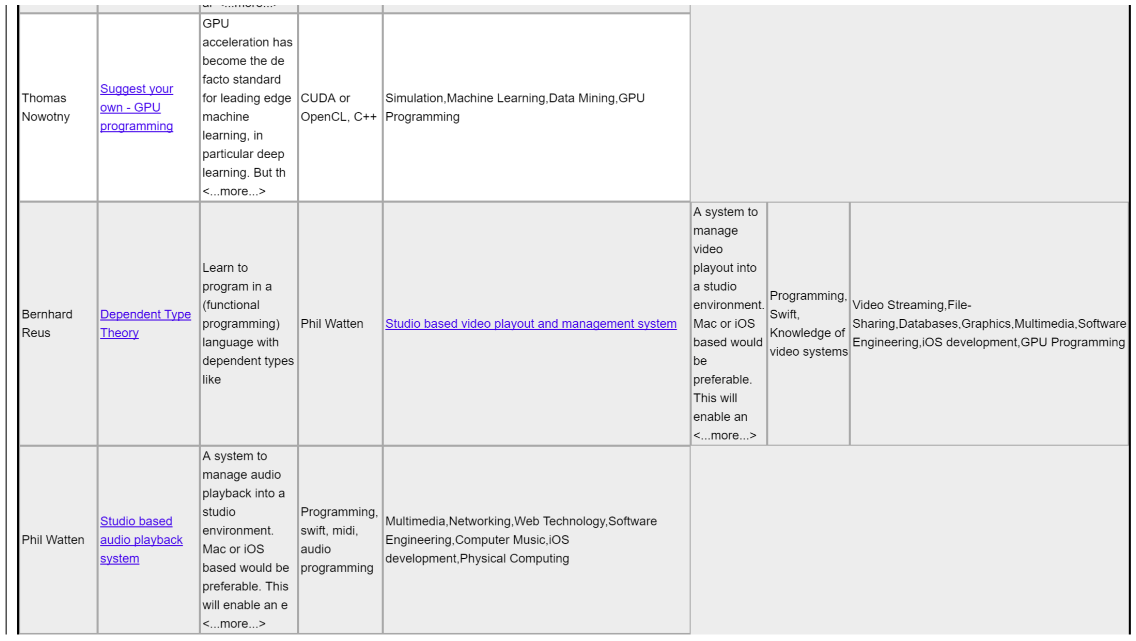
Figure 5: The "Student Options" drop-down menu.

A rather peculiar bug in the existing solution is drop-down menus seemingly being empty. While we can not be sure what cases the bug, one could speculate what the issue is. After so many minutes a user is automatically logged out of ITS authentication system. While they are not ITS authenticated, they are still authenticated on the website. As a result the user only has the guest privilege, therefore they are forbidden from accessing any user menus.

How was this issue improved upon?

The website now periodically re-authenticate with ITS while the session is still alive.

3.2.8 Issue - Tables



Thomas Nowotny	Suggest your own - GPU programming	GPU acceleration has become the de facto standard for leading edge machine learning, in particular deep learning. But th <...more...>	CUDA or OpenCL, C++	Simulation,Machine Learning,Data Mining,GPU Programming			
Bernhard Reus	Dependent Type Theory	Learn to program in a (functional programming) language with dependent types like	Phil Watten	Studio based video playout and management system	A system to manage video playout into a studio environment. Mac or iOS based would be preferable. This will enable an <...more...>	Programming, Swift, Knowledge of video systems	Video Streaming,File-Sharing,Databases,Graphics,Multimedia,Software Engineering,iOS development,GPU Programming
Phil Watten	Studio based audio playback system	A system to manage audio playback into a studio environment. Mac or iOS based would be preferable. This will enable an e <...more...>	Programming, swift, midi, audio programming	Multimedia,Networking,Web Technology,Software Engineering,Computer Music,iOS development,Physical Computing			

Figure 6: A broken table showing projects.

Another rather strange bug in the existing solution is occasional layout errors in project tables. The website seems to merge projects in the table, making some projects undiscoverable for students. There is no explanation as to why this occurs.

How was this issue improved upon?

This issue is not a problem on the new website, as pages are rendered using Laravel Blade. For more, read section 3.3.2.

3.3 New solution

The existing solution is written in vanilla PHP [10], HTML, CSS [44], and JavaScript. There is nothing wrong with this approach, but it limits how rapidly a developer can add and adapt features. Because of particular functional requirements, and as a personal preference, the new solution will be built using a full web-stack. Having this stack means the new solution will utilise multiple technologies for every aspect of the site.

3.3.1 Front-End Development

The front end consists of three main portions, document layout, document styling and document manipulation. Respectively the languages used for this are HTML, CSS, and JavaScript.

CSS

CSS is an excellent language but has some constraints. To overcome some of the constraints, a language named Sass will be used. As described at [11],

> ”Sass is the most mature, stable, and powerful professional grade CSS extension language in the world.”

Sass has many more features but outputs vanilla CSS, so it is a pre-processor for CSS. One major feature which makes Sass a valuable choice is variables. One of the functional requirements is to have easily configurable colours. Having variables will result in administrators being able to change the colours of the website with ease, not having to search through thousands of lines of CSS.

Sass also provides CSS vendor prefixes at compile time. In short, this ensures the CSS will be compatible with every browser without the developer having to remember to put them in.

JavaScript

JavaScript faces a whole heap of issues [25] when it comes to cross-browser compatibility. It even has issues with different versions of the same browser. To aide with these issues, a library named jQuery will be used. As described at [18],

> ”jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers...”

Using jQuery means more time can be spent programming functions that are helpful, and not having to worry if it works across every browser. Prototyping, writing features, and testing features will be a magnitude amount faster than vanilla JavaScript because jQuery has countless useful features. Some of the best features include a robust AJAX API, outstanding event handling, and top of the class DOM and CSS manipulation.

3.3.2 Backend Development

PHP

PHP [10] is the server-side programming language ITS servers use. Because of security concerns, there is no room for flexibility to use another language such as Ruby on Rails. PHP is useful for websites of all sizes, but vanilla PHP can become a hassle to manage. Although the new solution is not a huge website, it is quite complicated, with different types of users, filters when browsing projects, software layers and much more.

Because of the complexity of the project, and the productivity needed to complete it on time, a PHP framework will be incredibly useful. Many weeks have gone by looking for a best-fit framework for this solution, frameworks found to be potentially useful include Symfony [37], Yii [19], CakePHP [2] and Laravel [29].

Upon researching each framework, the last item on the list, Laravel, seems to be an ideal solution, and here's why. Out of the box, Laravel includes everything this project needs to be stable, functional and completed on schedule. Some of Laravel's best features used for the solution include, a robust routing API with authentication grouping, HTML templating, OpenSSL encryption, a rock-solid authentication API, database migrations, database transactions with rollbacks and a whole plethora of additional features. Having these features makes ticking off functional requirements a breeze.

HTML

HTML is a client-side language, however, a server can render HTML before delivering the file to the recipient. One of the most useful ways of using this technology is HTML templates. Vanilla PHP does not have any HTML template engine. However, every single framework mentioned above includes such engine without additional plugins. An example use would be iterating over each student to create a list. The template engine allows snippets of HTML to be used across the rest of the project. An example snippet would look something like this;

```
<li class="student">
  <p>{{ $student->first_name }}</p>
  <p>{{ $student->last_name }}</p>

  @if($student->last_login === null)
    <p>Never</p>
  @else
    <p>{{ $student->last_login }}</p>
  @endif
</li>
```

Figure 7: Example blade code snippet.

3.3.3 Functional Requirements

The functional requirements are broken down to different levels.

- **System Administrator -** A user who has the highest privilege. They have the ability change everything on the system.
- **Project Administrator -** A user who can change anything related to their education level. This will likely only be a maximum of two users, the final year student project administrator, and the master student project administrator.
- **Supervisor -** A user who can add projects to the system, accept or reject student offers, and email students.
- **Student -** A user who can send an offer to an administrator or propose a project.
- **Authorised User -** A user whom has logged in.
- **Guest -** Anyone who has a University of Sussex credentials. This is generally used for students to browse projects early.
- **Other -** Technical requirements unrelated to users, but related to the project.

ID	**Description**	**Optional**
	System Administrator	
FR 1	A system administrator should be able to add or remove a department.	Yes
FR 2	A system administrator should be able to add or remove an education level.	Yes
FR 3	A system administrator should be able to enable or disable user agent string collection.	Yes
FR 4	A system administrator should be able to enable or disable URL referral collection.	Yes
FR 5	A system administrator should be able to change the header logo.	Yes
FR 6	A system administrator should be able to enable or disable accessibility buttons.	Yes
FR 7	A system administrator shall be able to view the ”User agent stings” page.	Yes
FR 8	A system administrator shall be able to view the ”User feedback” page.	Yes
FR 9	A system administrator shall be able to view the ”System dashboard” page.	Yes
FR 10	A system administrator shall be able edit details of any user.	Yes
FR 11	A system administrator shall be able to add a user with any privilege.	Yes

	Project Administrator	
FR 12	A project administrator shall only be able to modify and view data according to their education level.	
FR 13	A project administrator shall be able to see when a user last logged in.	
FR 14	A project administrator shall be able to email any user.	
FR 15	A project administrator shall be able to change global parameters. The project year, the earliest selection date for students and the earliest selection date for supervisors.	
FR 16	A project administrator shall be able to add a new user of any privilege within their education level, except system administrators.	
FR 17	A project administrator shall be able to edit an existing user.	
FR 18	A project administrator should be able to edit or delete multiple students at a time.	
FR 19	A project administrator should be able to edit or delete multiple supervisors at a time.	
FR 20	A project administrator shall be able to amend supervisors arrangements. This includes said supervisors project load and whether or not they are accepting student offers.	
FR 21	A project administrator shall be able to browse all transactions and sort by transaction name or date.	
FR 22	A project administrator shall be able to browse transactions by project.	
FR 23	A project administrator shall be able to add a new topic.	
FR 24	A project administrator shall be able to amend the name of any topic.	
FR 25	A project administrator shall be able to import new students into the database with a CSV file.	
FR 26	A project administrator shall be able to check imported students in a test database before confirming.	
FR 27	A project administrator shall be able to email students grouped by project status.	
FR 28	A project administrator shall be able to export an archive at the end of the academic year. The archive includes which student was accepted for what project.	
FR 29	A project administrator shall be able to log in as any other user without knowing the user's authentication details. This function excludes other administrators (Project and system).	

FR 30	A project administrator shall also be considered as a supervisor. Therefore A project administrator shall be able to perform all supervisor actions.	
FR 31	The system shall be support multiple project administrators.	
	Supervisor	
FR 32	A supervisor shall be able to see their final year student project load.	
FR 33	A supervisor shall be able to see their master student project load.	
FR 34	A supervisor shall be able to see their student offers.	
FR 35	A supervisor shall be able to see projects proposed to them by a student.	
FR 36	A supervisor shall be able to 'undo' after they have accepted a student for a project. This action will also rollback the database to the state it was before the accept transaction, without effect other actions performed by other users.	
FR 37	A supervisor shall be able to identify whether an offer is from a final year or master student.	
FR 38	A supervisor shall be able to accept or reject a student's offer.	
FR 39	A supervisor shall be able to add a new project to the database.	
FR 40	A supervisor shall be able to browse their projects.	
FR 41	A supervisor shall be able to edit their projects.	
FR 42	A supervisor shall be able to see all accepted projects along-side the accepted student's name.	
FR 43	A supervisor shall be able to email all accepted, rejected or proposed students at once, or individually.	
FR 44	A supervisor shall receive a notification email when they receive a new offer.	
FR 45	A supervisor shall receive a notification email when a student proposes a project to them.	
FR 46	A supervisor shall be able to configure which notification emails they receive.	
	Student	
FR 47	A student shall only be able to see projects which are ***not*** archived, and where the supervisor's project load is greater than zero.	
FR 48	A master student shall only be able to see master projects.	

FR 49	A final year student shall only be able to see final year projects.	
FR 50	A student shall be able to see their project status.	
FR 51	Once accepted, a student shall be able to see the project they have been accepted for.	
FR 52	A student shall only be able to select a project if they have not already.	
FR 53	A student shall be able to propose a project to a supervisor.	
FR 54	A student shall be able to propose an open project (Where any supervisor can take the students project).	
FR 55	A student should have an option to share their selected/accepted/proposed project with other students on the ”Report by Supervisor” page.	
FR 56	A student shall not have any other privileges.	
	Authorised User	
FR 57	An authorised user shall be able to search projects.	
FR 58	An authorised user shall be able to browse projects on offer.	
FR 59	An authorised user shall be able to browse projects by supervisor.	
FR 60	An authorised user shall be able to browse projects by topic.	
FR 61	An authorised user shall be able to filter their search by title, description, supervisor, or topics.	
FR 62	An authorised user shall only be able to see relevant content on the ”Help” page.	
FR 63	An authorised user shall be able to see ”Report by Supervisor” page. This page shows every un-archived project a supervisor has, which students have been accepted and which student have selected said project.	
FR 64	An authorised user shall have any of the following privileges. Student, staff, supervisor, project administrator and system administrator.	
	Guest	
FR 65	A guest shall be able to log-in with their ITS user-name and password.	
FR 66	A guest shall be able to view the home page.	
FR 67	A guest shall be able to view the information page.	
FR 68	A guest shall be able to view the about page.	
FR 69	A guest shall be able to view the help page.	
FR 70	A guest shall be able to provide feedback using a form.	Yes

	Database	
FR 71	A transaction shall have a unique ID, transaction time, transaction type, project id, related supervisor and/or related student.	
FR 72	A user shall have a unique ID, first name, last name, username, email, last log-in date and access type ('student', 'staff', 'supervisor', 'admin').	
FR 73	A project shall have a unique ID, title, description, skills, status ('on-offer', 'withdrawn', 'student-proposed', 'archived'), supervisor ID and student ID.	
FR 74	A supervisor shall have a unique ID, title, contact type, project load and a take student boolean.	
FR 75	A student shall have a unique ID, registration number, programme, project status ('none', 'selected', 'proposed', 'accepted'), project ID and a share project boolean.	
FR 76	A topic shall have a unique ID and a unique name.	
FR 77	Prior to a topic being deleted, said topic shall be disassociated from all projects.	
FR 78	Prior to a user being deleted, said user shall be disassociated with all projects.	
	Other	
FR 79	A project shall have multiple topics.	
FR 80	A project shall have a single primary topic.	
FR 81	Final year and masters projects shall be stored in separate databases.	
FR 82	There shall be a single log-in for all users.	
FR 83	The website shall be cross-browser compatible with all major browsers. Browsers that must be compatible include includes Microsoft Edge, Google Chrome, Mozilla Firefox, and Apple Safari.	
FR 84	The website shall scale with screens of all sizes. The minimum supported resolution shall be 320x568 (A small screen mobile device in portrait mode).	
FR 85	The website shall use Sussex ITS authentication API.	
FR 86	The website shall be compatible with Sussex ITS servers and databases.	

3.3.4 Non-Functional Requirements

ID	Description
NFR 1	The server shall be written in PHP.
NFR 2	The front-end should be written in JavaScript.
NFR 3	The front-end should be structured with HTML.
NFR 4	The front-end should be styled with CSS.
NFR 5	The website should have University of Sussex branding.
NFR 6	The website should have easily configurable colours.
NFR 7	The website should be accessible for users visual impairments.
NFR 8	All administrator actions shall be organised into a ”Administrator hub”.
NFR 9	All supervisor actions shall be organised into a ”Supervisor hub”.
NFR 10	A user should be able to identify which version of the site they are on easily. (Final Year or Masters)

4 Design

4.1 Designing a Scalable Solution

The topic of scalability was discussed during a meeting, and the outcome was that the solution should be designed to support multiple departments. So this means designing a system not only to be used by the Informatics department but also any other department (from any school within Sussex), with the Engineering department being a prime candidate. When the original solution was being designed only Informatics undergraduate projects were planned for. This meant spending some time redesigning the solution, including the database, back-end and a small number of changes to the front-end.

4.2 Front-end Design

Designing a functional and attractive front-end is just as important as having a solid back-end. With undergraduate and postgraduate projects being converged onto a single website, it’s crucial that the user experience is great so students and supervisors won’t be confused.

Making sure users know which education level they are on is important, especially faculty, as they access both undergraduate and postgraduate. Having clear headings and a label in the main header at all times will make sure everyone always knows where they are.

It is important that users clearly know where they are, and that the website identifies itself with the university. The University of Sussex brand and content guide for web [28] has a collection of documents and media which should be used for digital media identifying with the university. The universities primary colourway will be used as an accent for the website, and according to [34], it is the universities most legible colourway as a background for text.

The design of the front-end was also inspired by the fifteen laws of user experience. For more on these laws, read [53].

4.2.1 Responsive Design

A vital feature for the front-end is to have a responsive design that scales for all screen sizes. I wanted to make sure the design is coherent between desktop and mobile devices. Many websites, such as Wikipedia, have a completely different design for desktop and mobile. Having different designs complicates the code base, and means a user will have to learn two different user interfaces.

Building a responsive design is not inclusive of a new header, it also includes content, moreover, tables. A major problem with the old website is that tables are quite a mess on mobile because they are not responsive. The content on the new website will scale to any screen size.

4.2.2 Animations & Transitions

The website includes animations and transitions throughout to make the user experience as enjoyable as possible. Unfortunately, it's quite hard to explain how the animations look or show them with a text document, however, I can explain why they are needed and how they come together to improve the user experience.

Granted [41] is not a scientific paper, nonetheless, the six guidelines from the article make complete sense. Guidelines 2, 4 and 5 are the most important to me, but I have tried to follow all of them. To quote the guidelines;

1. Animations shouldn't be an afterthought.
2. Animations must serve a purpose.
3. Animations must reflect the brand.
4. Animations shouldn't be the hero.
5. Animations must feel natural.
6. Animations must not waste time.

Following guideline 2 is needed to create a seamless experience throughout the website. The website uses transitions to soften harsh cuts, this can be seen in the help page and the system administrator dashboard. Both of these pages feature a tab system, and when a user selects a new tab, the transition between the previous and next tab's content is clear, minimising potential confusion to the user.

Following guideline 2 also helps provide context to the user. The website uses a method I like to call "Fake AJAX". This feature is used to create an illusion of dynamic content being loaded on static pages. An example of this is on the project page after a user submits a search, the current project table is replaced with a loading animation, this animation continues to play until the server responds to the search request. This provides a nice illusion but also provides instant feedback to the user.

Guideline 4, in short, is do not be too flashy. There are a few modern websites which use dramatic, disoriented and misguided animations and transitions. The main point of animations are to create a seamless experience, not a show.

Guideline 5, means making sure animations flow nicely and do not appear to be too stiff and robotic. Most animations use an easing function (Quadratic, Sine, Elastic, etc..) to help them appear more natural.

4.3 Back-end Design

I decided to also design for n amount of education levels, with undergraduate and postgraduate being the two implemented from the start. Although it might not be used, this means doctorate projects could also be added easily.

4.3.1 Localisation

Localisation is used to deliver websites in multiple languages without having to re-write the front-end in every language. Most programming languages have localisation built-in, PHP being one of those languages. A function similar to `get_locale_string("Welcome_Message")` which would be used to retrieve a string from a file. If an English user was on the site, the function would return ”Hello. Welcome to our website!”, and if the user was French, “Salut. Bienvenue sur notre site!” would be returned.

This solution only requires English, however we can exploit this functionality for separate departments and education levels. So we could have an Informatics ”language”, Engineering ”language” and so on for-each department. This can also be used for each education level. This may seem complicated, especially if there were ten departments, but this can be organised neatly into sub-folders.

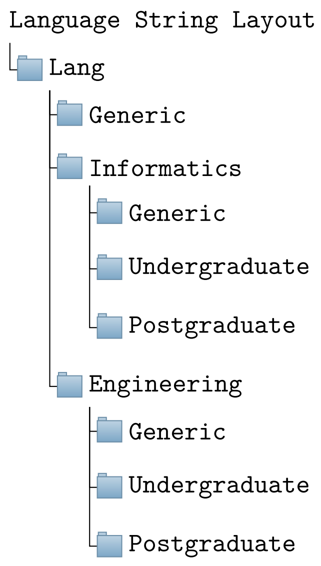


Figure 8: The layout of localisation files.

5 Security

5.1 Authentication

The first step of authentication for the application is LDAP[22]. LDAP, an abbreviation for Lightweight Directory Access Protocol, is the authentication method used by ITS at the University of Sussex. This method allows us to identify who is a member of the university, which department they belong to, and which level of education they are. Once a user has authenticated with ITS (External authentication), the server checks in the internal database to check whether or not a user is present. If a user is not in the internal database, they will be treated as a guest.

5.2 Privileges

Privileges is a system to make sure only certain users can perform certain actions. For example, a user without the system administrator privilege should not be allowed to perform system administrator tasks. Each user has one or more privilege(s) assigned to them, stored as a MySql SET in the database. The list of privileges include;

- **Student** - An undergraduate or postgraduate student, a user who can select projects.
- **Staff** - A staff member, *generally* reserved for faculty who are no longer participating.
- **Supervisor** - A supervisor, a user who can create projects.
- **Admin_ug** - An undergraduate administrator, a user who can change undergraduate parameters and data.
- **Admin_pg** - A postgraduate administrator, a user who can change postgraduate parameters and data.
- **Admin_system** - A system administrator, a user who can change system parameters and settings.

5.3 Cross-Site Request Forgery Protection

Cross-site request forgery [51], abbreviated as CSRF or XSRF, is a common malicious exploit on websites. The exploit allows an unauthorised user to perform commands as an authorised user.

A typical technique of defending from CSRF attacks to use a synchroniser token, known as a CSRF token. The CSRF token is a secret and unique value that must be sent with every form request to the server. This token will then be checked by the server for every type of request, except GET. Without this token, an attacker is unable to place the correct token in their requests to authenticate themselves.

A downside to using a token arises when a user has multiple tabs open in their browser, as a unique token is generated with each request. An easy way to circumvent this issue is to only generate a token per session. This means once a user is authenticated, they will have the same token for their entire session.

Another way of storing a CSRF token is as a cookie, however, this has a disadvantage to a meta field. The disadvantage to storing the token as a cookie is a potential for a session hijacking [52] (Cookie hijacking) to occur. If a users session was hijacked by an attacker, the attacker would have the users' CSRF token. If the victim was a system administrator this could be disastrous.

The token is stored as a meta field in the head section of every HTML file sent from the server. The CSRF token meta field in the HTML head looks like this;

```
<head>
    <meta name="csrf-token" content="{{ csrf_token() }}">
    ...
</head>
```

Figure 9: Example CSRF token code snippet.

The token is also included in every HTML form using a hidden input field. It is also included in every AJAX request, this is achieved by setting a HTTP header `X-CSRF-TOKEN` to the token value.

5.4 Client/Server Side Form Validation

Each and every form on the website has tailor-made client-side validation. Client-side validation is useful for restricting input to types such as email or numbers and setting minimum and maximum input lengths. In addition to restrictions, it is also beneficial for reducing server load, as invalid requests would not be sent to the server, reducing the number of times the server has to perform form validation.

In spite of advantageous client-side form validation can be, it is far from perfect. Validation can easily be circumvented in any browser using basic developer tools to edit HTML and JavaScript. This is where server-side validation becomes utilitarian.

Laravel provides plenty of validation techniques, including form validation, known as a ”Form request”. A Laravel form request has two basic methods to override, `rules()` and `authorize()` . The rules method lets custom validation logic to be defined, the quickest way to explain this is with a snippet;

```
public function rules(){
    return [
        'username' => 'required|unique:users|max:7',
        'password' => 'required|min:10',
    ];
}
```

Figure 10: Example custom validation logic.

Without covering every rule, let us cover some of the most powerful and frequently used. Required is self-explanatory, so is min and max. The unique rule is robust, it is equivalent to unique on a MySql server, but it can happen before trying to insert into the database. This functions by querying the column in the database and cross-checking it with the form field. This can be an expensive operation, but so are database errors.

The `authorize()` method is simple but useful. It is just another way to check whether a user is authorised to perform the action. If a user is not authorised, the form will never validate.

5.5 Input Sanitisation

MySql is one of the recommended databases for Laravel, thus support for the database is pre-eminent. As a result, Laravel automatically sanitises queries sent to the database using middleware, which is covered next.

5.6 Security Middleware

Middleware is a general term for software that acts as a bridge between two separate systems. In Laravel, middleware is a convenient mechanism for filtering HTTP requests. Middleware is used extensively in the web application for multiple use cases, including security. As middleware literally holds the application together, let us cover some critical middleware that specially written for this application;

- **AdminPrivilegeCheck** - This middleware is to circumvent a privilege oversight. The issue is supervisors can perform undergraduate and postgraduate tasks, however, they can also be a project administrator. Say a user is a supervisor and an undergraduate project administrator, the user would be able to perform postgraduate project administrator tasks because they inherit the postgraduate privilege from being a supervisor. This middleware circumvents this oversight by cross-checking the session education level with the administrator's privileges, regardless if they are a supervisor.
- **DepartmentCheck** - Checks to see if a department is set, if not, redirects to department selection.
- **EncryptCookies** - Encrypts cookies with bcrypt [50] to hide the cookies content.
- **TrimStrings** - Used for server-side form sanitisation.
- **TrustProxies** - Checks the file type and rejects the request if the file type is not supported by the server.
- **VerifyCsrfToken** - Checks for CSRF tokens in POST requests and rejects the request if the token is not valid or present.
- **Admin** - Checks to see if the user is any type administrator.
- **ProjectAdmin** - Checks to see if the user is a project administrator.
- **SystemAdmin** - Checks to see if the user is a system administrator.
- **SupervisorOrAdmin** - Checks to see if the user is an administrator or supervisor.
- **Student** - Checks to see if the user is a student.
- **Supervisor** - Checks to see if the user is a supervisor.

The user privilege middleware is vital for routes, which will be covered and explained in the backend development section.

5.7 UUID

In general, databases will store sequential IDs which auto increment, therefore ID's increment from 1 to *n*. This could have security implications; Say there is a website with a list of users, you look at a few users and see the URLs are similar, e.g. `www.example.com/user?id=250`, `www.example.com/user?id=251`, `www.example.com/user?id=252`, etc. This would imply that the IDs in the database are auto incrementing. This method is not inherently bad, however, if an attack was to happen, it may be easier to increment through every user record.

This is where UUIDs[12] (Universally Unique Identifier), also known as GUIDs (Globally Unique Identifier), can be useful. UUIDs are unique 128-bit numbers identifiers. There have been multiple specifications of UUID, the solution uses version 4, which generates the identifies with secure random numbers.

The first benefit of UUIDs is that they can be generated offline, usually a server has to query the database to find the next available ID, however, this is not the case for UUIDs.

The second benefit of UUIDs is that it is practically impossible to find another one in the database, as a result, targeted attacks are nullified.

A downside to UUIDs is that they are large, four times larger than the 4-bit integer identifier that would have been used. None-the-less, this is not really an issue because the database won't contain any table with millions of rows. Another downside is, unlike incremental IDs, UUIDs can clash with each other. According to [20], the odds of this happening are around $0.00000000006(6*10^{-11})$.

As a side note, the solution uses ordered identifiers, which utilises time-stamp first UUIDs. Ordered IDs can be stored and indexed in a database as efficiently as incrementing IDs, so no need to worry about performance [21].

6 Development

6.1 Architecture

6.1.1 MVC

Model view controller (MVC) is a very common architectural pattern used in a range of applications. MVC separates the application into three key sections. This pattern decouples the three sections allowing classes and methods to have a clear place in the code-base.

Laravel, the PHP framework being used, encourages the MVC pattern. The only difference with ”web MVC” is that, in a sense, there is also another controller, the front-end controller (Which uses JavaScript). Therefore, not only does the user interact with the backend controller, the front-end controller (Limited to AJAX calls) interacts with the backend controller.

- The model is the pivotal component of the pattern. It expresses the application's behaviour in terms of the problem domain, independent of the user interface. It directly manages the data, logic and rules of the application. More on models in the next section.
- A view can be any output representation of information. A view is not restricted to being an HTML page, but it can also a little snippet of HTML, such as a table or form, typically used with AJAX. There is no view to controller ratio, which may result in numerous views utilising the same controller.
- The back-end controller accepts input (Generally from HTTP requests) and converts it to commands for the model or view.
- The front-end controller sends and retrieves data from the view and backend-controller.

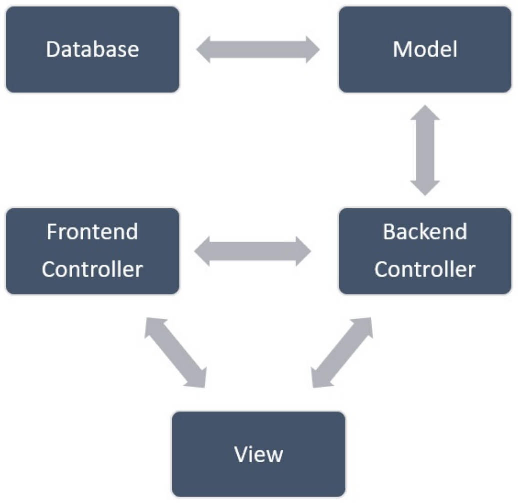


Figure 11: A diagram of the web MVC architecture.

6.1.2 Model

Models are, in essence, a definition of a table in the database. The methods a model generally contain are used to help describe an instance of the model, for example, the user model contains methods such as `isProjectAdmin()` and `isStudent()` . Models never directly interact with the controller, only the database. Let us look at the models in the application;

- **User** - The user model is used for every type of user, a base class if you will. This model can check all properties of a user, and has a relationship with student, supervisor and project models.
- **Student** - The student model handles all student, undergraduate and postgraduate. It has methods for retrieving the student's favourite projects, second supervisor (Marker), project and spoofing the student's name (If they have chosen too).
- **Supervisor** - The supervisor model is quite large. It has methods for setting and getting project load, student take, email preference and more. It also includes methods for retrieving all of the supervisor's projects, their second supervising students (Students they are the second supervisor too), all of their student proposal and their student selections.
- **Project** - The project model contains just a few methods, including retrieving the primary topic, students with this project selected and the student who was accepted to undertake this project. The project model has a relationship with the topic model.
- **Topic** - The topic model contains three methods, projects (Projects with this topic), project on offer and data list (A HTML5 data list used for searching).
- **ProjectTopic** - The project topic model is used solely as a pivot for projects to topics.
- **Transaction** - The transaction model contains only one method, as the data is only ever retrieved or stored, never manipulated.
- **Mode** - The mode model is a singleton model, as there should only ever be one row in the mode table. It contains methods for retrieving the start date and project year.

- **UserAgentString -** The user agent string model contains only one method, which gets the table name.

6.1.3 Controller

6.1.4 MVC Downsides

One downside to MVC is sometimes not knowing whether to implement a method in the model or the controller. An example of this is a method named `calculateSecondSupervisors()`, which assigns second supervisors to students, this method is in the admin controller. It's a method only an administrator can perform, but it affects the student and supervisor models, not the admin model. Let's say the method was in the supervisor model, then the supervisor model would be altering the student model, which is absolutely forbidden.

6.2 Backend

6.2.1 Laravel

Laravel [29] is the glue that holds the whole application together. This section will cover how Laravel has helped, how it has been used, and which features were used the most.

Laravel is one of the most popular[3] PHP frameworks around. One of the reasons Laravel is so popular is because of how robust of a framework it is. It includes APIs for everything that was needed to write the application, front-end and back-end.

Eloquent ORM

Eloquent ORM[31] (Object-relational mapping) is an API included with Laravel. Eloquent ORM provides a simple active record[49] implementation for working with a database. As said before in the model section, each database table has a corresponding Model which is used to interact with that table (Multiple tables in the case of the user and student model).

As a result of this, in most cases, we can query and modify data in the table using ORM, instead of writing MySql queries. Let us look at a small example to understand it better;

```php
ProjectTopic::where('project_id', '47a4-asd3')
    ->where('topic_id', 'ds3d-34fd')
    ->delete();
```

(a) Eloquent ORM.

```sql
DELETE FROM `informatics_project_topics_ug`
    WHERE `project_id` = '47a4-asd3' AND 'topic_id' = 'ds3d-34fd';
```

(b) Regular MySql statement.

[3]There is no real measure of popularity. However, popularity can be measured by the web activity of the framework (Open source project, blogs, news, meet-ups, etc..).

This is a simple case, however, we can see how useful ORM can be. Firstly, it can automatically resolve the table name from the model, remember there are undergraduate and postgraduate tables, as well as tables for each department. Secondly, if this was user input, it would be automatically sanitised. The larger the query, the more helpful ORM can be. Using this API really speed up the development process.

6.2.2 HTTP Web Routes

Routes are a way of mapping a URI (Uniform Resource Identifier) to a method, this is known as routing. Laravel provides a powerful API for defining custom routes.

Routes can be attached to any HTTP verb, GET, POST, PUT, PATCH, DELETE or OPTIONS. What makes Laravel's routes so powerful is an injection method known as route model binding. Let's look at a regular route definition for updating a project

```
Route::patch('projects/{project-uuid}/edit', 'ProjectController@update');
```

This patch method would then call the update method in the project controller, which would then have to search the project table for a project which has a matching id to 'project-uuid'. However, with route model binding, we can define a route with a model in the URI which Laravel would detect and inject the model, which can be directly accessed by the controller. This is beneficial in many ways, one being that the code footprint in controllers will be reduced. Another being a lower chance of bugs occurring, because a trivial line such as `Project::where('id', '=', 'project-uuid')` would not have to be planted at the top of every single method in every single controller.

I have included a list of routes used by the application, it is an e-file available at *RouteList.txt*.

6.2.3 Mailables

Every type of email in Laravel is represented as a Mailable class. You can think of them as a model for an email. Mailables would generally vary greatly, but for this solution, every email needs the same three public properties, a supervisor, student and a project. These properties will be assigned via a constructor when a new mailable is instantiated.

We can use Blade to create an email template, then it's as easy as serialising the models, plugging the values into the view, then sending the email to its recipient. The list of emails being sent include;

- Student accepted for a project (*Sent to student*).
- Student rejected for a project (*Sent to student*).
- Student proposed a project (*Sent to supervisor*).
- Student selected a project (*Sent to supervisor*).
- Student unselected a project (*Sent to supervisor*).

6.2.4 Application Middleware

Middleware was covered from a security perspective before, let's look at them from an application perspective.

- **Language** - As mentioned before, the way localisation is used in the system is quirky. The language middleware ensures the user's locale is persistent throughout the user's session.
- **Required cookie** - Sets all the cookies required throughout the website. This is useful because it ensures cookies are set in every request, meaning null checking `!empty($_COOKIE['CookieName'])` for cookies is not necessary.
- **Set department** - Checks the request query for a department, then sets the department, only if it exists in the system.
- **Set education level** - The same as the previous middleware, however, it also checks if the authenticated user is authorised to access said education level. This and the former middleware are helpful for links in emails. Having the department and education level in the URL means the user can be redirected to the correct page.
- **User agent string collector** - A simple middleware that collects the user agent string from the request and stores it in the database.

6.3 Front-end

The front-end encompasses the whole look, feel and interaction with the end user.

6.3.1 Laravel Blade

Laravel Blade is an HTML templating engine. It allows views to inherit, yield and include other views. This is convenient because we can use a single layout template for most views, a layout template would look something along the lines of this;

```
<!DOCTYPE html>
<html lang="{{ app()->getLocale() }}">
  @include ('partials.html-head')

  <body>
    @include('partials.header')
    <div class="main-content">
      @yield('content')
    </div>
  </body>

  @include('partials.footer')
</html>
```

Figure 13: An example code snippet of Laravel Blade.

This template can now be used for all kinds of views such as 'browse projects' and the homepage. It includes the HTML head which contains all the meta-data needed for browsers. And the header needed for navigation. The `@yield('content')` is where the body of the page would be included.

6.3.2 SASS

I decided to write all of the styling code myself to give the website a unique look. As stated before SASS was used for styling the website.

SASS is an expressive language, and the three features that were used most are import, mixins and variables. None of which are to be found in plain CSS.

Import allows the styling code for a somewhat large project to stay manageable, allowing the styling to be spread between 25+ different files, with just a single, main file, importing them. With plain CSS, this would have been a monstrous 7K lines long file.

Mixins are groups of CSS declarations that you can be reused throughout the styling. Mixins also accept parameters which make them incredibly flexible. These are also favourable for keeping a consistent style throughout.

Variables allows us to define any value to a variable. This may sound obvious, but with plain CSS it can be a chore to have consistent styling.

6.3.3 JavaScript

An issue I personally see on the modern web is huge file sizes, generally down to monstrous JavaScript files. I kept the transfer sizes down to a minimum by only including JS files which are necessary for that page. This entails having separate JS files for some views, such as `tab-view.js`. This file is only required on pages which have a tab view (Help and system dashboard). Having separate JS files can save up to 680Kb per request on this website.

Asynchronous JavaScript

AJAX, an acronym for Asynchronous JavaScript And XML, is a technique used by many websites to send and receive data to a server without having to refresh the page. AJAX is supported by nearly all browsers since it uses the browsers built in XMLHttpRequest function.

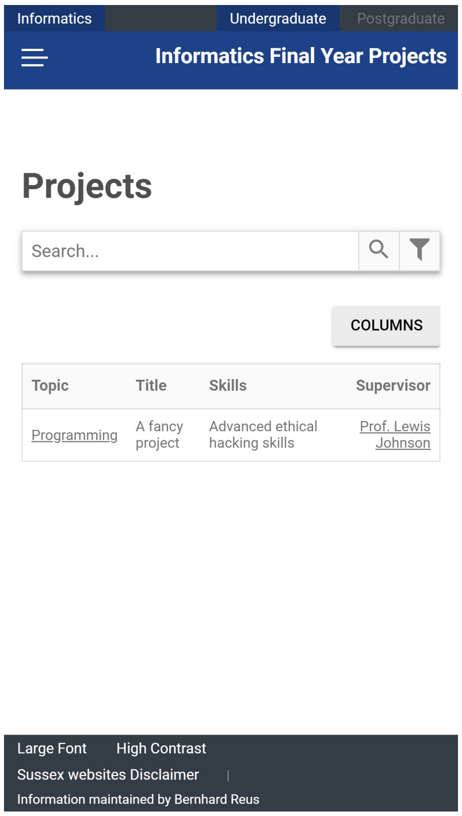
AJAX has been sprinkled throughout the new website, mainly for supervisor and administrator pages. Having AJAX on most supervisor and administrator pages allows faculty to complete tasks a whole lot quicker than a full page refresh every time they want to perform an action.

6.4 Mobile

This section covers features that were implemented or influenced by mobile devices.

6.4.1 Project Previews

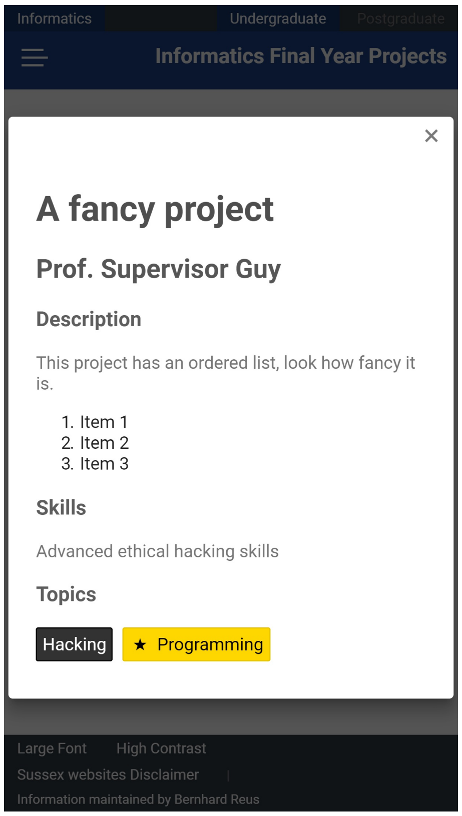
A small feature added for mobile devices (Note it still works on a desktop computer), which was inspired by a similar feature on Instagram, is ”project preview”. Project previews were added to help solve the issue of screen real estate on small screen devices. The issue is, on mobile, a user can not see all the details of a project in the table. However, with a project preview, a user can simply hold the project row for 250ms to view a preview of the project.


Informatics
Undergraduate
Postgraduate
Informatics Final Year Projects
Search...
COLUMNS
Large Font
High Contrast
Sussex websites Disclaimer
Information maintained by Bernhard Reus

Projects

Topic	Title	Skills	Supervisor
Programming	A fancy project	Advanced ethical hacking skills	Prof. Lewis Johnson

(a) Browse projects page.


Informatics
Undergraduate
Postgraduate
Informatics Final Year Projects
×
Hacking
★ Programming
Large Font
High Contrast
Sussex websites Disclaimer
Information maintained by Bernhard Reus

A fancy project

Prof. Supervisor Guy

Description

This project has an ordered list, look how fancy it is.

1. Item 1
2. Item 2
3. Item 3

Skills

Advanced ethical hacking skills

Topics

(b) Project preview.

6.5 Other

6.5.1 Project Management

A team of one still needs to keep organised in order to keep track of progress and issues. To manage the development side of this project I decided to use an extremely popular website named GitHub [9], which uses the even more popular revision control system named Git [17].

GitHub most useful feature is version control in a repository, which allows changes to the project to be tracked and described with a commit. GitHub also allows issue tracking and feature cards. Issue tracking is used to keep track of bugs with tags, such as Internet Explorer, Firefox, Chrome, PHP, etc and an associated priority. This is a great alternative to spamming "**CHROME BUG #4956: Something here doesn't work!**" comments throughout the code base. The feature cards feature was used throughout the development to keep on track of progress according to the project plan, the feature cards are compiled from the requirements table, so there were hundreds of features to keep track of, and this made it easy.

6.5.2 Documentation

For PHP I used a tool named phpDocumentor [43]. PhpDoc converts standard documentation to be converted into a web page, allowing a developer to easily navigate classes, methods and properties.

Simply open `Final-Year-Database/docs/index.html` in a browser to see. JavaScript and the rest of the code base are mostly documented with inline comments.

7 Accessibility

The University of Sussex has a wide range of students with a wide range of special requirements. As this solution is designed to be scalable to x amount of users, it is important to make sure the website is accessible to everyone.

The Accessible Rich Internet Applications (WAI-ARIA) 1.1 [47] specification explains in great detail how to make sure a website is usable with assistive technologies. I used this as a reference while following W3C's ARIA overview [45]. Accessibility is not inclusive of ARIA support, it is also about following the Web Content Accessibility Guidelines (WCAG), because of this I will also be following these guidelines [46]. As an auxiliary, I will also be using Mozilla's accessibility documentation [24] to ensure all types of users are accommodated for.

7.1 ARIA

The ARIA specification has five keywords that must be followed to ensure the specification is followed correctly, the keywords being; **MAY**, **MUST**, **MUST NOT**, **SHOULD**, and **SHOULD NOT**. They are described in detail in RFC2324 [5].

7.1.1 The Roles Model

ARIA 1.1 Section 5

The roles model is a way of describing the role of a DOM element to assistive technologies. This is possibly the most important section of the specification and will be followed to a tee. A role is described as follows;

> ”Main indicator of type. This semantic association allows tools to present and support interaction with the object in a manner that is consistent with user expectations about other objects of that type.”

So this means, the roles model makes it easy for users to identify what the role of an element is, how it can be interacted with, and in most cases, what the outcome will be. The roles model is implemented throughout the website for modals, tables, forms, buttons and every other element detailed in the specification.

A great use of the role model is with the mobile hamburger, button which is a div element with a role=”button” attribute.

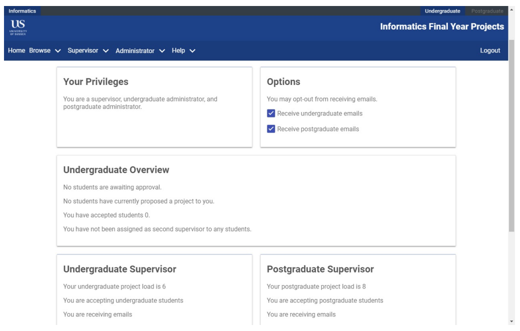
[language=HTML]

```html
<div class="mobile-menu" role="button">
    <...>
</div>
```

Figure 15: An example of how the ARIA role model can be implemented.

7.2 Supplementary Accessibility Options

The website includes extra accessibility options that some users may find useful. This includes a large font option as well as high contrast options. Both of these options can be toggled by buttons in the footer of the website. The large font option scales all text relative to its previous size to ensure the website keeps the same look. The high contrast option turns borders and fonts from grey to black.



Your Privileges

You are a supervisor, undergraduate administrator, and postgraduate administrator.

Options

You may opt-out from receiving emails.

- [✓] Receive undergraduate emails
- [✓] Receive postgraduate emails

Undergraduate Overview

No students are awaiting approval.

No students have currently proposed a project to you.

You have accepted students 0.

You have not been assigned as second supervisor to any students.

Undergraduate Supervisor

Your undergraduate project load is 6

You are accepting undergraduate students

You are receiving emails

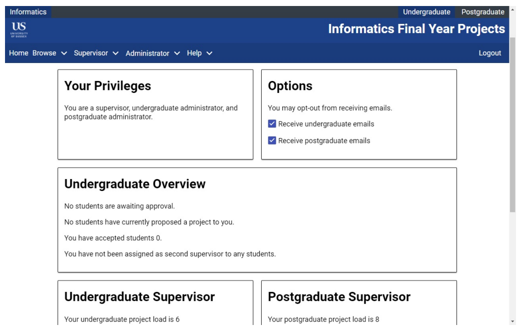
Postgraduate Supervisor

Your postgraduate project load is 8

You are accepting postgraduate students

You are receiving emails

(a) The new website with accessibility options turned off.



Your Privileges

You are a supervisor, undergraduate administrator, and postgraduate administrator.

Options

You may opt-out from receiving emails.

- [✓] Receive undergraduate emails
- [✓] Receive postgraduate emails

Undergraduate Overview

No students are awaiting approval.

No students have currently proposed a project to you.

You have accepted students 0.

You have not been assigned as second supervisor to any students.

Undergraduate Supervisor

Your undergraduate project load is 6

Postgraduate Supervisor

Your postgraduate project load is 8

(b) The new website with accessibility options turned on.

7.3 Validating Accessibility

It is all good and well to follow guidelines and specifications, but with a website this large, it can be easy to miss a role here or a there. This is why the website's accessibility needs to be validated with a tool. I will be using Paypal's automated accessibility testing tool (AATT) [32]. This tool allows validation to be automated, checking every single page in less than a minute. The catch is, AATT only checks WCAG and can not check pages which require authentication, nevertheless it was used for home and help pages.

Fortunately, Google Chrome includes an accessibility audit tool [13] named Lighthouse [15] which can be used to validate accessibility on any page on the website. Lighthouse will be mentioned again in the testing section as it isn't exclusive to accessibility testing. Lighthouse has been used to manually validate every single page on the website, a score of 100 was achieved on most pages, with the score dropping to 98 at it's lowest.

8 Testing

8.1 Automated Browser Testing

Laravel Dusk [30] is an expressive browser automation and testing API. Dusk uses utilises ChromeDriver [42], which is an open-source tool for testing web apps. Dusk's API lets us simulate a real user by mimicking user inputs and interactions.

The type of tests one would write for Dusk are unlike unit tests. Dusk tests are somewhat of an interaction flow containing every click and key press a user would. As an example, a navigation test. This navigation test first simulates a user logging in, as a user with every privilege, the test then hovers (Mouse overs) over every single drop-down in the navigation bar to check whether or not the contents of the said drop down are visible. If the navigation becomes broken for whatever reason, this test will catch it and fail. This kind of testing is impossible with conventional unit testing.

8.2 Testing Common Malicious Attacks

A data scientist named Max Woolf compiled a huge list of stings, aptly name ”The Big List of Naughty Strings” [23]. These strings are likely to cause errors or issues with servers and databases. Some of the strings in the list have caused issues for notable tech companies, such as Twitter Inc., in the past. Each string from the 683 line long list has been tested on against a user input form using Laravel Dusk.

9 Evaluation

The project has not yet been deployed on a server at the University of Sussex, so that is probably the most critical requirement failed. The reason for this is a migration from ITS controlled servers to the department's own server, I have no role in this situation. The current plan is to deploy the website in the coming weeks.

A somewhat minor, but annoying oversight is the system administrator. The solution was adapted to support multiple departments after planning and designing were well underway. There was suddenly a need for a new user, the system administrator. However, as mentioned before, the user tables are split per department, so for example, informatics_users and engineering_users. So where does the system administrator go? A user independent from any department. A solution would be to create a new table, however a table for a single user is a little excessive, so the solution is the system administrator can be in any user table.

Another incomplete functional requirement is FR 85. Which is to allow users to log in with their Sussex ITS log-in details. This requirement is only partially incomplete, as guest log-in is possible. The problem is not knowing how to get details about the user using LDAP, therefore it can not be determined which department they are from, and which education level they are. This will be fixed sometime after the deadline.

Regarding the rest of the requirements, the project has implemented all of the core and optional requirements to a reasonable standard.

10 Conclusion

In conclusion, the project is a success and will hopefully help hundreds of students and supervisors over many years. I achieved all of the objectives, and the only features incomplete are neither requirements nor relevant (Except deploying the application on a server).

Future extensions would be few, but there are some things which would make for a more polished website. I am by no means a creative or a designer, and a designer could probably design more welcoming and polished visuals for the website. Another extension could be to flesh out the browsers tests because there is always an untested edge case.

I managed to finish the website on time, however, I could have saved a tremendous amount of time if I had used one of the hundreds of CSS frameworks available online, such as Bootstrap [1]. I decided to write all of the styling code myself to give the website a unique look, in hindsight, this was probably a mistake. Using a framework would have saved countless hours styling trivial elements such as check-boxes, toggles, drop-downs, navigation, etc. Not only would it of saved me time, it would have provided a consistent style across browsers, as with my custom styling, elements such as drop-downs will look slightly different across browsers.

I also should have used a JavaScript framework, such as Vue.js [54] (Which is recommended for a Laravel project, with support out the box). JavaScript frameworks can take some time to understand and write for, however, in the long run, it would have saved me a lot of time writing repetitive and mundane functions.

Lastly, thank you Dr.Reus for being an amazing supervisor.

References

[1] Maintained by the core team with the help from contributors. Bootstrap · the most popular html, css, and js library in the world. `https://getbootstrap.com/`, 2017. [Online; accessed 20-April-2017].

[2] Inc. Cake Software Foundation. Cakephp - build fast, grow solid. `https://cakephp.org/`, 2017. [Online; accessed 03-Nov-2017].

[3] Oracle Corporation. Mysql. `https://www.mysql.com/`, 2018. [Online; accessed 29-Mar-2018].

[4] Tammy Everts. Page bloat: The average web page size is more than 2mb comments feed. `https://www.soasta.com/blog/page-bloat-average-web-page-2-mb/`, 2015. [Online; accessed 07-Nov-2017].

[5] IETF Internet Engineering Task Force and Scott Bradner. Key words for use in rfcs to indicate requirement levels. `hhttps://tools.ietf.org/html/rfc2119`, 1997. [Online; accessed 09-Feb-2018].

[6] IETF Internet Engineering Task Force and Larry Masinter. Hyper text coffee pot control protocol (htcpcp/1.0). `https://tools.ietf.org/html/rfc2324#section-2.3.2`, 1998. [Online; accessed 09-Feb-2018].

[7] JS Foundation. webpack. `https://webpack.js.org/`, 2018. [Online; accessed 31-Mar-2018].

[8] The Apache Software Foundation. Apache http server project. `https://httpd.apache.org/`, 2018. [Online; accessed 29-Mar-2018].

[9] Inc. GitHub. Github. `https://github.com/`, 2017. [Online; accessed 01-Sep-2017].

[10] The PHP Group. Php: Hypertext preprocessor. `http://php.net`, 2018. [Online; accessed 29-Mar-2018].

[11] Chris Eppstein Hampton Catlin, Natalie Weizenbaum and individual contributors. Sass: Syntactically awesome style sheets. `http://sass-lang.com/`, 2015. [Online; accessed 03-Nov-2017].

[12] M. Mealling IETF Internet Engineering Task Force, P. Leach and Microsoft. Key words for use in rfcs to indicate requirement levels. `https://tools.ietf.org/html/rfc4122`, 2018. [Online; accessed 29-Apr-2018].

[13] Google Inc. Audit rules google chrome. `https://github.com/GoogleChrome/accessibility-developer-tools/wiki/Audit-Rules`, 2017. [Online; accessed 22-Dec-2017].

[14] Google Inc. Final year project database. `https://goo.gl/qzPGSd`, 2017. [Online; accessed 06-Nov-2017].

[15] Google Inc. and individual contributors. Lighthouse | tools for web developers | google developers. `https://developers.google.com/web/tools/lighthouse/`, 2018. [Online; accessed 03-Feb-2018].

[16] NGINX Inc. Nginx | high performance load balancer, web server, reverse proxy. `https://www.nginx.com/`, 2018. [Online; accessed 29-Mar-2018].

[17] Over 1100 individual contributors. Git. `https://git-scm.com/`, 2017. [Online; accessed 01-Sep-2017].

[18] The jQuery Foundation. jquery. https://jquery.com/, 2017. [Online; accessed 03-Nov-2017].

[19] Yii Software LLC. Yii php framework: Best for web 2.0 development. http://www.yiiframework.com/, 2017. [Online; accessed 03-Nov-2017].

[20] lutz. guid - how unique is uuid? - stack overflow. https://stackoverflow.com/questions/1155008/how-unique-is-uuid, 2009. [Online; accessed 29-Apr-2017].

[21] Nikos M. The differences between int and uuid in mysql. https://stackoverflow.com/questions/30461895/the-differences-between-int-and-uuid-in-mysql, 2018. [Online; accessed 08-Apr-2018].

[22] Microsoft. Lightweight directory access protocol. https://msdn.microsoft.com/en-us/library/aa367008(v=vs.85).aspx, 2018. [Online; accessed 29-Apr-2018].

[23] Max Woolf (@minimaxir). The big list of naughty strings is a list of strings which have a high probability of causing issues when used as user-input data. https://github.com/minimaxir/big-list-of-naughty-strings, 2017. [Online; accessed 03-Feb-2018].

[24] Mozilla and individual contributors. Accessibility | mdn. https://developer.mozilla.org/en-US/docs/Web/Accessibility, 2017. [Online; accessed 22-Dec-2017].

[25] qdirks chrisdavidmills Juggernaughtt Mozilla, EdwardB. Handling common javascript problems. https://developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/Cross_browser_testing/JavaScript, 2017. [Online; accessed 03-Nov-2017].

[26] Jordi Boggiano Nils Adermann and many community contributions. Composer. https://getcomposer.org/, 2018. [Online; accessed 29-Mar-2018].

[27] Inc npm. npm. https://www.npmjs.com/, 2018. [Online; accessed 31-Mar-2018].

[28] University of Sussex. University templates and logo - brand and content guide for web and print. http://www.sussex.ac.uk/brand/templates, 2017. [Online; accessed 06-Oct-2017].

[29] Taylor Otwell. Getting started with laravel. https://laravel.com/docs/, 2015. [Online; accessed 06-Oct-2017].

[30] Taylor Otwell. Browser test. https://laravel.com/docs/5.6/dusk, 2018. [Online; accessed 20-Apr-2018].

[31] Taylor Otwell. Eloquent: Getting started. https://laravel.com/docs/5.6/eloquent, 2018. [Online; accessed 28-Apr-2018].

[32] Inc. PayPal Holdings. Github - paypal/aatt : Automated accessibility testing tool. https://github.com/paypal/AATT, 2017. [Online; accessed 22-Dec-2017].

[33] David Thomas & Bernhard Reus. Final year project database. https://www.informatics.sussex.ac.uk/courses/csaiproj09/index.php, 2011. [Online; accessed 20-Oct-2017].

[34] Color Safe. Color safe - accessible web color combinations. http://colorsafe.co/, 2017. [Online; accessed 06-Oct-2017].

[35] Alberto Savoia. How much unit test coverage do you need? - the testivus answer. http://www.artima.com/forums/flat.jsp?forum=106&thread=204677, 2010. [Online; accessed 03-Nov-2017].

[36] Mark Seemann. Code coverage is a useless target measure. http://blog.ploeh.dk/2015/11/16/code-coverage-is-a-useless-target-measure, 2015. [Online; accessed 03-Nov-2017].

[37] SensioLabs. Symfony, high performace php framework for web development. https://symfony.com/, 2017. [Online; accessed 03-Nov-2017].

[38] Helen Sharp, Yvonne Rogers, and Jennifer Preece. *Interaction design: beyond human-computer interaction*, pages 35–38. Wiley, 4th edition, 2016.

[39] Wolfgang Skala. Gantt charts with the pgfgantt package. https://www.overleaf.com/latex/examples/gantt-charts-with-the-pgfgantt-package/jmkwfxrnfxnw, 2017. [Online; accessed 01-Nov-2017].

[40] SMARTDOME. Smartdome constructions. http://www.smartdome.si/, 2017. [Online; accessed 07-Nov-2017].

[41] José Torre. 6 animation guidelines for ux design - 6 animation guidelines for ux design. https://blog.prototypr.io/6-animation-guidelines-for-ux-design-74c90eb5e47a, 2017. [Online; accessed 10-Feb-2018].

[42] Unknown. Chromedriver. https://sites.google.com/a/chromium.org/chromedriver//, 2018. [Online; accessed 20-Apr-2018].

[43] Unknown. phpdocumentor. https://www.phpdoc.org/, 2018. [Online; accessed 08-Apr-2018].

[44] W3C. Cascading style sheets. https://www.w3.org/Style/CSS/, 2018. [Online; accessed 24-Apr-2018].

[45] World Wide Web Consortium (W3C) and individual contributors. Wai-aria overview | web accessibility initiative (wai) | w3c. https://www.w3.org/WAI/intro/aria, 2006. [Online; accessed 22-Dec-2017].

[46] World Wide Web Consortium (W3C) and individual contributors. Web content accessibility guidelines (wcag) 2.0. https://www.w3.org/TR/WCAG20/, 2008. [Online; accessed 22-Dec-2017].

[47] World Wide Web Consortium (W3C) and individual contributors. Accessible rich internet applications (wai-aria) 1.1. https://www.w3.org/TR/wai-aria-1.1/, 2017. [Online; accessed 22-Dec-2017].

[48] Neil Walkinshaw. *Software Quality Assurance - Consistency in the Face of Complexity and Change*. Springer, University of Leicester, United Kingdom, 2017.

[49] Wikipedia and individual contributors. Active record pattern. https://en.wikipedia.org/wiki/Active_record_pattern, 2018. [Online; accessed 28-Apr-2018].

[50] Wikipedia and individual contributors. bcrypt. https://en.wikipedia.org/wiki/Bcrypt, 2018. [Online; accessed 02-Apr-2018].

[51] Wikipedia and individual contributors. Cross-site request forgery. https://en.wikipedia.org/wiki/Cross-site_request_forgery, 2018. [Online; accessed 02-Apr-2018].

[52] Wikipedia and individual contributors. Session hijacking. https://en.wikipedia.org/wiki/Session_hijacking, 2018. [Online; accessed 02-Apr-2018].

[53] Jon Yablonski. Laws of ux. www.lawsofux.com, 2017. [Online; accessed 10-Feb-2017].

[54] Evan You. Vue.js. https://vuejs.org/, 2017. [Online; accessed 20-April-2017].

11 Appendix

Notes

The body of this document (From section 1 to 10) contains 10250 words. The project plan LaTeX uses a template adapted from [39]. The existing solution currently used by the department of Engineering and Informatics can be found at [33].

The repository hosted on Github will be made public sometime after the deadline, on 30/04/2018 16:00. You are free to browse the code and commits here, `https://github.com/LewisJohnson/Final-Year-Database`.

11.1 Error 418

You might not know of the HTTP error status code 418. RFC 2324, section 2.3.2 [6] clearly states;

> ”Any attempt to brew coffee with a teapot should result in the error code ”418 I’m a teapot”.”

So, if a user visits the URI ”`/teapot`” they will be welcomed with an error message and an animation of a teapot pouring a cup of tea.

11.2 Deployment

This section covers how to deploy the application on any Apache or Nginx server.

Prerequisites

You will need a few applications before getting started;

- An Apache *or* Nginx server with terminal access (Apache tested and recommended) [8] [16]
- PHP >=7.1.3 (PHP 7.2.3 tested and recommended) [10]
- Composer [26]
- MySql *or* MarinaDB [3]

If you are running Linux, now is a good time to make sure existing software and sources are up to date.

```
sudo apt-get update
sudo apt-get upgrade
```

11.2.1 Step 1 - Moving files

Copy the folder `Final-Year-Database` from the base folder of the e-uploads to any directory on your server.

11.2.2 Step 2 - Installing Dependencies

Navigate to the `Final-Year-Database/app` folder and open a terminal/command prompt window. Type the following command

```
composer install --no-dev
```

Now everything we need to get the application up and running is installed. Please note that this will install the bare minimum needed to get the application running.

11.2.3 Step 3 - Configuring the Environment

You will find a file named `.env` in the `Final-Year-Database/app` directory. Here you can setup the environment for the application. Let's cover the environment line-by-line;

Application

- `APP_NAME` - Internal use, do not change.
- `APP_ENV` - The application environment, this should be set to production.
- `APP_KEY` - The encryption key used throughout the Laravel framework for encrypting cookies and database fields.
- `APP_DEBUG` - If the application is in debug mode, this should be set to false. Debug mode shows detailed error pages that can help debug issues.

- APP_URL - The URL in which `Final-Year-Database/app` resided.
  For example, `https://www.informatics.sussex.ac.uk/projects/`.

  Database
- DB_CONNECTION - The database driver to use, this should be set to mysql.
- DB_HOST - The IP address of the database.
- DB_PORT - The port the database is running on.
- DB_DATABASE - The database name to use.
- DB_USERNAME - The username to authenticate with the database.
- DB_PASSWORD - The password to authenticate with the database.

  System
- BROADCAST_DRIVER - Internal use, do not change.
- CACHE_DRIVER - Internal use, do not change.
- SESSION_DRIVER - Internal use, do not change.
- QUEUE_DRIVER - Internal use, do not change.
- FILESYSTEM_DRIVER - Internal use, do not change.

  Email
- MAIL_DRIVER - The mail driver to use, this should be set to smtp.
- MAIL_HOST - The IP address of the mail delivery host, this will probably be `smtp.sussex.ac.uk`.
- MAIL_PORT - The port mail delivery host, this will probably be `587`.
- MAIL_USERNAME - The Sussex ITS username to authenticate with, for example `hk823`.
- MAIL_PASSWORD - The Sussex ITS password to authenticate with.
- MAIL_ENCRYPTION - The encryption method to use for sending emails, this should be set to `tls` if you are using ITS.
- MAIL_FROM_ADDRESS - The email address to spoof when sending emails, this will likely be `noreply@sussex.ac.uk`.
- MAIL_FROM_NAME - The name which appears when someone receives an email

For more information about possible values and other variables, please read the Laravel documentation.

11.2.4 Step 4 - Configuring the Server

The application directory is owned by our system user, and is readable but not writable by the web server. This is correct for the majority of application files, but there are few directories that need special treatment. Specifically, wherever Laravel stores uploaded media (Think student import) and cached data, the web server must be able not only to access them but also to write files to them.

```
sudo chgrp -R www-data /var/www/html/Final-Year-Database/app
sudo chmod -R 775 /var/www/html/Final-Year-Database/app/storage
```

11.2.5 No Terminal Access?

It is possible (but impractical) for the application to be deployed without terminal access. If you are in this situation, please contact me.

11.3 Maintaining the Application

This section covers how to maintain the application and create patches. Ensure you have followed the deployment section prior to continuing.

Prerequisites

In addition to the prerequisites prior, you will also need;

- Npm [27]

11.3.1 Step 1 - Installing Dependencies

Navigate to the `Final-Year-Database/app` folder and open a terminal/command prompt window. First we will have to type the following command

```
composer install
```

This command is different than before as it will also install development dependencies needed to make changes to the application. Secondly we need to install JavaScript dependencies using NPM, we can do this with the command

```
npm install
```

This will install a monstrous 975 dependencies, all listed in `app/package-lock.json`. The defined dependencies (Dependencies which are defined in `package.json`) are

- **@shopify/draggable** - A tiny library for dragging and swapping DOM elements. This is used to drag project topics.
- **cross-env** - Allows bash scripts to be run on all operating systems seamlessly.
- **jquery** - The jQuery library.
- **jquery-confirm** - Used for confirmation dialogues and modals.
- **laravel-mix** - Provides a fluent API for defining Webpack build for Laravel application. Also includes SASS/SCSS complication, source maps and versioning.

- **less.js** - Compiles LESS.
- **less-loader** - Compiles LESS.

11.3.2 Step 2 - Learn Webpack

Webpack [7] (Stylised as 'webpack') is a module bundler. It packs multiple JavaScript files into bundles so you can split the code base instead of having a single huge file. It can also compile JavaScript with babel, which ensures the scripts can be ran on any browser by compiling newer JS syntax (Such as ECMAScript 2018) to older syntax (Such as ECMAScript 2015). Not only does it bundle JavaScript, but it can also bundle and compile other assets such as Plain CSS, SASS, LESS and more.

You will find the webpack definition in `webpack.mix.js`. You will need to add any new assets to this build file. It is easy to understand and make additions. The Laravel documentation includes most things you'll need, but the webpack documentation goes into more depth.

11.3.3 Step 3 - Redeploy

Now you have changed all the code you want, run:

```
npm run prod
```

Now everything is ready. If you made changes without copying the code to another folder, the changes will become live immediately.

11.4 System Configuration

Here are some configuration files which you may want to change.

- `Final-Year-Database/app/resources/assets/jsconfig.js` - Here you can change the JavaScript configuration (Mostly unimportant).
- `Final-Year-Database/app/storage/app/configsystem.json` - Here you can change the departments and education levels.
- `Final-Year-Database/app.env` - Here you can configure the app, database and mail settings.

11.5 Meeting Log

3.10.2017 - **Meeting 1**

All of Dr.Reus' supervised students, including myself, met to discuss our chosen projects. What was expected from the project and how much effort it will require was outlines. We also individually planned bi-weekly meetings.

Outcome: I will meet with Dr.Reus in his office every other Monday at 2:30 PM.

9.10.2017 - **Meeting 2**

I meet with Dr.Reus to discuss the requirements for the project. We looked through the existing solution to see which features were critical in the new solution, this also included going through administrator and supervisor features. We also talked about potential barriers with ITS (Mainly outdated software and security restrictions.) and how these issues might be overcome.

Outcome: An early requirements document, which can be polished later to create a detailed document.

23.10.2017 - **Meeting 3**

I meet with Dr.Reus to get feedback on the project proposal and to comb over some of the client requirements again. While discussing the project proposal, we concluded that the current timetable is not realistic. We also concluded that the project proposal submitted was not sufficient in quality.

Outcome: The project proposal will be rewritten in a more professional tone. Rewriting the proposal includes creating a new timetable. Once completed, I will email the document for additional feedback.

30.10.2017 - **Meeting 4**

I meet with Dr.Reus to get feedback on the reworked project proposal and timetable. The feedback was positive with only very minor changes needed to be made for a fault-free document. I also asked a bunch of questions that were needed to make a better requirements document.

Outcome: A refined, near-complete requirements document.

13.11.2017 - **Meeting 5**

I meet with Dr.Reus to show him the first "working" version of the website. This was mainly to receive feedback in regards to the visuals side of the website, rather than features.

Outcome: Reduce the white space on desktop and use more of the screen space. Additionally the project card title font is too small, adjust the font sizes to convey importance.

27.11.2017 - **Meeting 6**

I meet with Dr.Reus to ask a few questions, mainly to clarify requirements.

Outcome: Consider other departments using the system. HTML editor for description in projects. Description should have a minimum length (Configurable).A project will have a marker (Second supervisor). Make a system which matches 2nd supervisors. An admin should be able to quickly match a 2nd supervisor to a project.

06.02.2018 - **Meeting 7**

I meet with Dr.Reus to show him that I had Laravel working on my personal ITS web space, an alternative route if we can not get a server from ITS or the department. I also showed new features including, system dashboard, project preview, table column selection and dynamic pagination.

Outcome: I have to focus on testing the system, making sure features actually work, and stay working.

20.02.2018 - **Meeting 8**

I meet with Dr.Reus to discuss automatic second supervisor (Marker) assignment.

Outcome: A set of rules on how to assign students to second supervisors according to a few variables.

Missing meeting logs.

20.04.2018 - **Meeting 9**

I meet with Dr.Reus to get feedback on my draft report. The overall quality seemed to be okay but grammar mistakes were plenty, with inappropriate English (Brilliant, Amazing, Incredible) for a technical report.

Outcome: Tips on how to improve my report and potentially achieve a higher grade.

11.6 Project plan

The project plan is split into two separate plans, Autumn and Spring.

11.6.1 List of activities

Activities in red are critical.

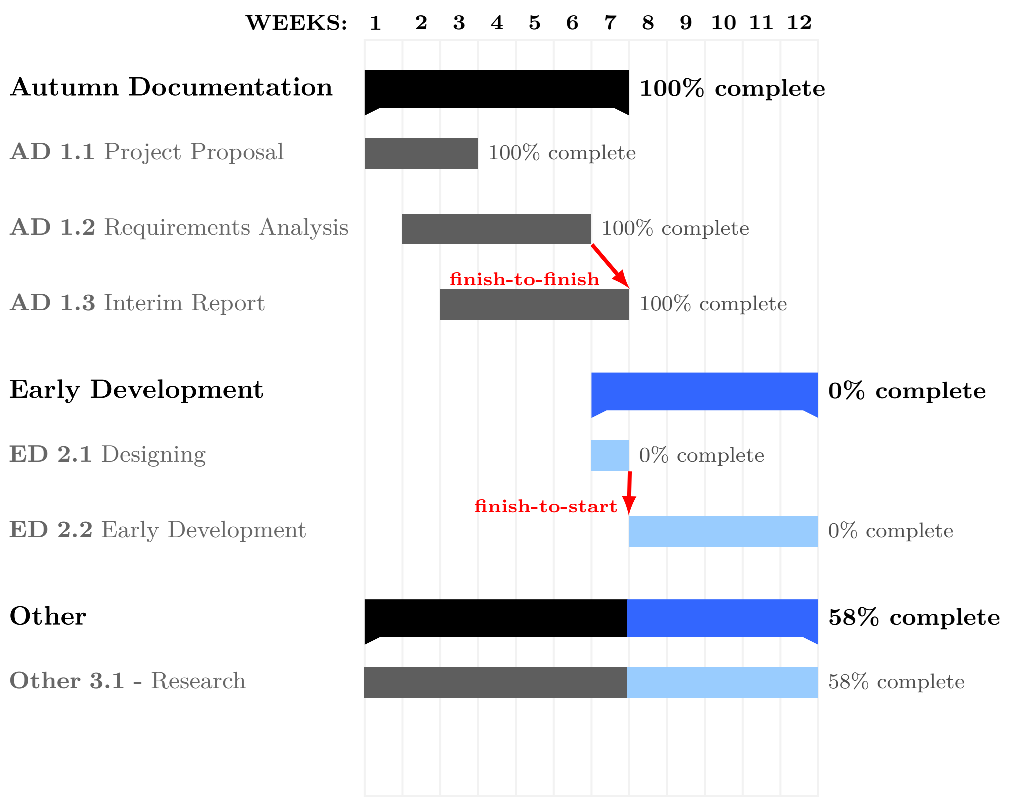
1. **Autumn documentation** - This includes the project plan and the project timetable.
   (a) **Project proposal**
   (b) **Requirements Analysis** - The requirements analysis is a separate activity because it's critical to the whole project. Making sure I create a refined, detailed and through requirements analysis will take some time.
   (c) **Interim Report** - The entire writing process for the interim report.
2. Early Development
   (a) **Designing** - This includes wire-framing the website, choosing colours, and thinking about user experiences for all users. Books [48] and [38] will be used throughout this activity to make sure software quality and user accessibility are as best as they can be.
   (b) **Development** - While most of the heavy development tasks will be done in the Spring term, some minor development tasks can be completed beforehand. Tasks include converting the database schema to SQL queries and writing some front-end code.
3. Other

   (a) **Research** - Throughout the entire autumn term, I will be researching potential server frameworks, reading user experience books. Every outcome from the research should be set in stone by the end of week 12.

4. Spring Documentation
   (a) **Project Presentation** - This activity includes creating a poster for the poster event and creating a presentation for the short talk.
   (b) **Draft Report** - The entire writing process for the draft report.
   (c) **Final Report** - The entire writing process for the final report.
5. Development
   (a) **Programming** - This includes writing all the HTML, CSS JavaScript, and PHP. This also encompasses writing the code documentation.
   (b) **Unit Tests** - This project will not be following the test-driven process. Instead all critical functionality will have unit tests. There will not be a code coverage target for this project, for more please read [35] and [36].
   (c) **User Testing** - Testing the software manually by myself and with the client.

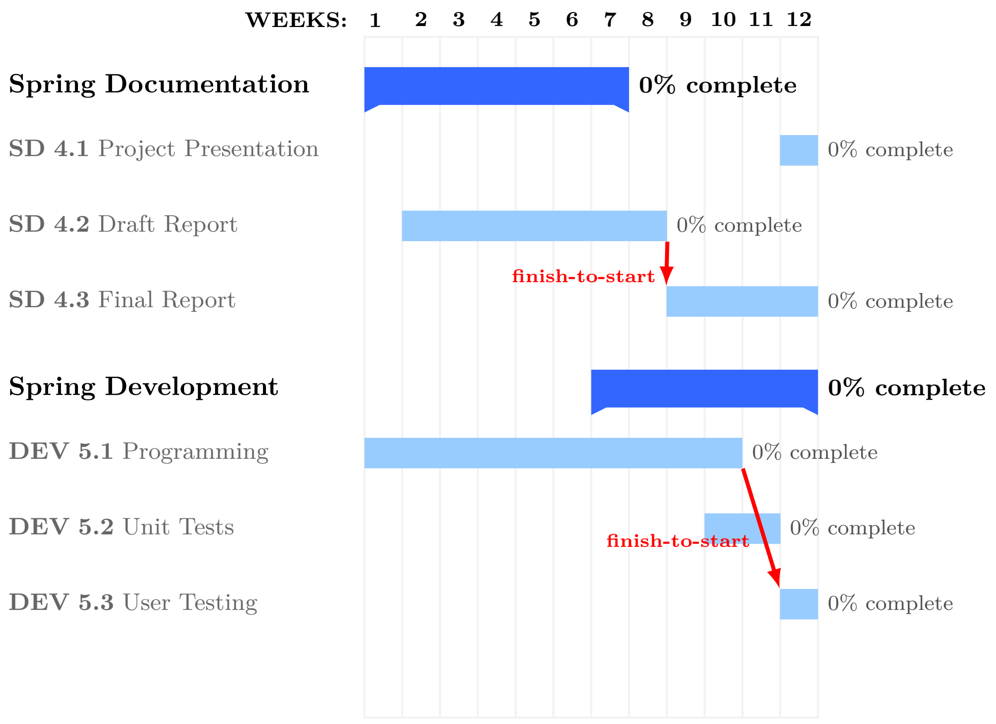
11.6.2 Autumn project plan

For week to date link, see figure 17a.



11.6.3 Spring project plan

For week to date link, see figure 17b.



11.7 Appendix - Figures

Autumn Term

- **Week 1** - 25th September 2017
- **Week 2** - 2nd October 2017
- **Week 3** - 9th October 2017
- **Week 4** - 16th October 2017
- **Week 5** - 23rd October 2017
- **Week 6** - 30th October 2017
- **Week 7** - 6th November 2017
- **Week 8** - 13th November 2017
- **Week 9** - 20th November 2017
- **Week 10** - 27th November 2017
- **Week 11** - 4th December 2017
- **Week 12** - 11th December 2017

(a) The Autumn term weeks.

Spring Term

- **Week 1** - 5th February 2018
- **Week 2** - 12th February 2018
- **Week 3** - 19th February 2018
- **Week 4** - 26th February 2018
- **Week 5** - 5th March 2018
- **Week 6** - 12th March 2018
- **Week 7** - 19th March 2018
- **Week 8** - 9th April 2018
- **Week 9** - 16th April 2018
- **Week 10** - 23rd April 2018
- **Week 11** - 30th April 2018
- **Week 12** - 7th May 2018

(b) The Spring term weeks.