

ME4 FINAL REPORT

ME4 Project Allocation Process and Website/Software

Thushaan Rajaratnam

CID: 00939666

Supervisor: Dr Ambrose Taylor

Department of Mechanical Engineering

Imperial College London

Date: 6 June 2019

Word Count: 11972/12000

ABSTRACT

This report outlines the project carried out to re-design the process and website used for allocating final year projects to final year undergraduate and master's students at Imperial College's Mechanical Engineering department. The main objective set by the author and the project supervisor was to build a website that enhanced the experience of matching projects for students, staff and the program coordinator (who was also the author's project supervisor). Literature research conducted revealed the inadequacy of re-designing the process, due to which the project focused on building a new website which would allow staff to upload projects and subsequently select students who had viewed and selected their desired projects. The report details the methodology used to host the website as well as code its front-end (display-end) and back-end (processing-end). In addition to also describing the features added as well as its security aspects, the testing performed on the website was elaborated to conclude that a more effective methodology should be devised. Using platforms such as Laravel, AJAX and bootstrap enabled a rapid development time frame as well as a more secure website to be developed. The testing also revealed future work that could be added onto the website to enhance the user experience, which can be done more easily due to this report and the standard coding frameworks used. The next step of the project would be to implement the developed website onto Imperial College's system, as detailed further in the report, and then allow future cohorts at the university to use this system for allocating final year projects. The project was deemed to be a success in that it achieved the main objective of building a website that enabled the project allocation process for all involved stakeholders: students, staff and the project coordinator.

CONTENTS

Abstract.....	2
1 Introduction	4
1.1 Background.....	4
1.2 Objectives	5
1.3 Report Structure.....	5
2 Literature Review	6
2.1 Project Allocation Process	6
2.2 Website Development.....	7
3 Project Allocation Website - Technology	9
3.1 Technology Stack	9
3.2 Project Database	16
3.3 Search Engine: Algolia.....	18
3.4 Security	19
4 Project Allocation Website – Features.....	21
4.1 Authentication	21
4.2 Creating, Deleting and Updating projects	25
4.3 Project Viewing	26
4.4 Project Selection and Ranking.....	28
4.5 Student Selection and Project Matching by Staff.....	28
4.6 Super Admin Panel.....	30
5 Testing	31
5.1 Sample Data	31
5.2 User Feedback.....	32
5.3 Vulnerability Testing.....	36
5.4 Device Compatibility Testing.....	38
6 Discussion.....	39
6.1 Success & Failures of Project.....	40
6.2 Implementation by University	41
6.3 Future Work	43
7 Conclusion	45
8 References.....	46

1 INTRODUCTION

1.1 BACKGROUND

During the final year of Imperial College's MEng and MSc Mechanical Engineering course, students must choose a Final Year Project (FYP) (Taylor, 2018). Currently, to do this, students use a website system developed in coordination with the Life Sciences department. At the beginning of the academic year, the system is updated with projects (approximately 200) from various staff after which students rank their preferences. Supervisors then choose their preferred students based on students' own selections. Next, the course coordinator (also the supervisor of this project) undertakes the process of manually exporting information from the website onto an excel sheet. The data is then manipulated to get a table of projects and their respectively allocated students. Finally, the coordinator informs students of their allocated projects.

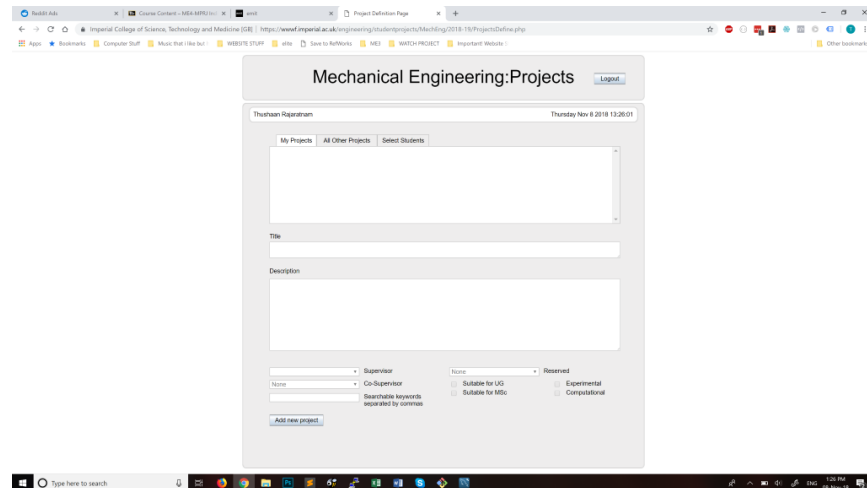


Figure 1: Current ME4 project allocation process and website/software

The students often complain about the current system in place for the allocation process of projects (see Figure 1). Many do not like the system's user experience and user interface (UX/UI) and sometimes do not get their desired project – this could possibly lead to poor student satisfaction (Leggett, 2018). This project is an opportunity to redesign the website experience for students, staff and the project coordinator to view, choose and allocate the projects. The project involves creating a new website with better UX/UI as well as data manipulation capabilities to facilitate the selection process for all involved users.

1.2 OBJECTIVES

To achieve the final goal of improving users' experience in the selection process, specific objectives were agreed between the author and the project supervisor as measurable milestones. The project began in October 2018 and concluded in June 2019, with the following objectives being pursued:

- Create a prototype ME4 FYP allocation website based on current solution in place such that all required functions are still achieved. Such functions include creation of Student/Staff accounts, creation of FYP projects, selection and allocation of the projects.
- Get feedback in the form of multiple surveys for students and staff for improvements of the website.
- Modify the prototype to improve the UX/UI based on feedback. Modify the prototype to improve the allocation process based on feedback.

These objectives and their progress are discussed in further detail in the report.

1.3 REPORT STRUCTURE

This final report is a continuation of 2 previous reports written on this project: the quality plan report (Rajaratnam, 2018) and progress report (Rajaratnam, 2019). The report describes how the work was carried out in order to achieve the project. After the literature review, the report describes the technology used to build the website in terms of the technology stack, database and user registration and security details. The report then describes the different features that were built using the technology stack mentioned, and how they improve the user experience with the website. The subsequent Testing section elaborates on how the results of the project were measured to evaluate if the objectives were successfully met, and to what extent. The results of the overall project are then discussed and expanded into suggestions for future work. The conclusion of the report summarizes the overall achievements of the project and the report itself.

2 LITERATURE REVIEW

A literature review was carried out in order to research existing methodologies of allocating projects to students in other institutions as well as different techniques of developing a website. Since the project was mainly focused on building the website, the literature review needed was brief but provided a solid framework for approaching the languages and templates used in terms of coding.

2.1 PROJECT ALLOCATION PROCESS

Abraham et al. (Abraham, 2007) investigated a method known as the Student-Project Allocation problem (SPA) which attempted to optimise the number of students getting their most preferred choice of projects using statistics. The paper framed SPA as a two-sided matching problem, where there would be 2 inputs A and B (A being the number of students and B the number of projects). The investigation attempted to optimally match members of A to members of B, depending on a range of criteria. These criteria could involve specific subject streams or student grades as preferred by the supervisors or the project co-ordinator.

A revelation of the study was that participants involved in two-sided matching problems should not interfere with the allocation manually by contacting each other or making ad hoc arrangements. This meant that students and staff should not interact with each other at all if the system were to use algorithmic matching of projects to students. The paper also concluded that the matching system should aim for stability within the sets of students and projects, such that two participants (i.e. a student and a project) that are not matched remain with their original selections without being forcefully matched to each other (Abraham, 2007).

The study used two linear-time algorithms to find a stable matching for SPA problem. One of the algorithms optimised for student choices over supervisors, and the other optimised for supervisor choice of students over the students' own choices.

Romero-Medina et al. (Romero-Medina, 1998) explored the system used in the United States' National Resident Matching Program (NRMP) which has been allocating 30,000 graduating medical students and residents annually to their first hospital posts, based on the participants' choices. This was found to be one of the most effective large scale centralised matching algorithms that solved the two-sided matching problem. More allocation systems

were also used and explored in other educational institutions including universities in Spain and Scotland, as well as schools in Asian countries such as Indonesia and Singapore.

A paper by Anwar et al. (Anwar, 2003) researched an internal system within University of York's Department of Computer Science to allocate student projects which weighs both the student and lecturer choices. Whilst having a preference hierarchy of student, lecturer and projects in that descending order, the study investigated the use of programming techniques to find a stable matching between students and projects with constraints such as balancing the supervision load among lecturers with equanimity.

A recurring conclusion of literature was the ineffectiveness of using algorithms to match students and projects beyond a certain degree. As Manlove et al. (Manlove, 2008) concluded, an optimal factor of 50% was the highest achievable effective allocation between students and projects. It was deemed that further investigation was required to determine the existence of a better algorithm which would outperform the results yielded by manual decision making between students and staff over which projects are allocated to what student.

Since literature on project allocation systems indicated that the effectiveness of algorithmic matching is limited, it was decided that the current system being used by the Mechanical Engineering Department would be left in place. i.e. the 80% students getting their first choice of projects was still a better outcome than any conceivable algorithm.

2.2 WEBSITE DEVELOPMENT

There is a lot of literature on best practices to develop a website for different applications. Since the principal use of this project was for users to be able to easily visualize projects, students and staff as well as select them, there was emphasis on researching data storing techniques and user experience guides.

Research by Otwell et. al (Otwell, 2019) suggested that the best way to build a website back-end is by basing it on existing frameworks such as Laravel. Laravel provides a base layer of infrastructure which makes back-end website functions more easily codable, in that it comes pre-packaged with a lot of server functionalities that would otherwise have to have been coded from the ground up. The Laravel documentation was extensively used during the development of the website since it leveraged the framework heavily (Otwell, 2019). As

discussed in Section 3.1.1, the framework allowed for much faster prototyping while maintaining good website practices. The document described how Laravel prevents the need for the author to manually code best-practice features such as authentication, encryption, cookies and sessions into the website.

In order to improve the experience of users visualizing and interacting with website information, research on front-end development was carried out. 2 principal sources of literature that provided objective results were documentation on front-end frameworks such as Twitter Bootstrap and jQuery (js.foundation, 2019) (Mark Otto, 2019). The thesis by Otto et al. discussed the use of front-end frameworks to provide standard templates of code that enabled website content and graphics to be easily adapted to different applications and screen sizes. The papers also included references to online front-end libraries which provided functions that were useful for managing components such as the general layout of the page, forms, and animations. The libraries were used extensively by the author when designing the website's user experience and interface (UX/UI).

A literature review of the search functionality within websites was also performed, which revealed a paper by Dessaigne et al. (Nicolas Dessaigne, 2019) discussing the Algolia search framework. The paper elaborates how a third-party service such as Algolia can be incorporated into a website to produce highly configurable search rankings within the author's website's database. In addition to enabling searching through the list of projects, students and staff, the paper also discusses ways in which the website's search function can be improved to understand and predict user searches intelligently. The author has recommended this paper as a useful source for carrying out potential future work on the project as entailed further on in the report.

An online document (Maatwebsite, 2019) detailing information about the Laravel-Excel plugin available was referred to when building a feature on the website that allowed data to be exported as Microsoft Excel spreadsheets. The Laravel-Excel plugin documentation discusses the feature principally to be used by the "superadmin" end user (project coordinator Ambrose Taylor, primarily to easily hand over information to Imperial College's Undergraduate office as a course requirement). The documentation was highly relevant in the initial quick setup of the package, which then allowed for more advanced tasks such as the ability to export the required information in different formats such as CSV (Comma Separated

Variables), XLS (Excel Spreadsheet) and specifying the headers of each of the tables, custom value formatting and so forth.

The documentation on website development was used to then start building the website as detailed in the next section.

3 PROJECT ALLOCATION WEBSITE - TECHNOLOGY

The main objective of the project was to build a website, which required considerations of its technology stack, database management software as well as the methodology used to onboard users and look after their information's authentication and security. This section describes the steps involved in developing a website from scratch, and the frameworks used to specifically suit a project selection and allocation application.

3.1 TECHNOLOGY STACK

A website contains and displays all requested information to the user's screen. All this information (data) is stored and manipulated in the back-end of the website, whilst it is displayed on the front-end. The data is physically stored on a server, which is a remote computer that receives requests to store and send data. The data is stored within a database, which in the case of the FYP website is manipulated by PHP scripts and the SQL (Structured Query Language) relational database management language (Nixon, 2014). Scripts are instructions that enable the server to serve requests for information from the database, which it returns to the front-end (display) of the website in HTML (Hyper Text Markup Language), CSS (Cascading Style Sheets) and JavaScript code. Whilst HTML elements (text, tables, images) contain the content of the website pages (webpages), JavaScript affects the behavior and relations of these contents, and CSS affects their appearance and layout (Nixon, 2014).

This entire process is the technology stack and most websites on the internet have their own unique stack (Wodehouse, 2018). Both the local and live technology stacks for the case of the FYP website are shown in Figure 2 and Figure 3 respectively.

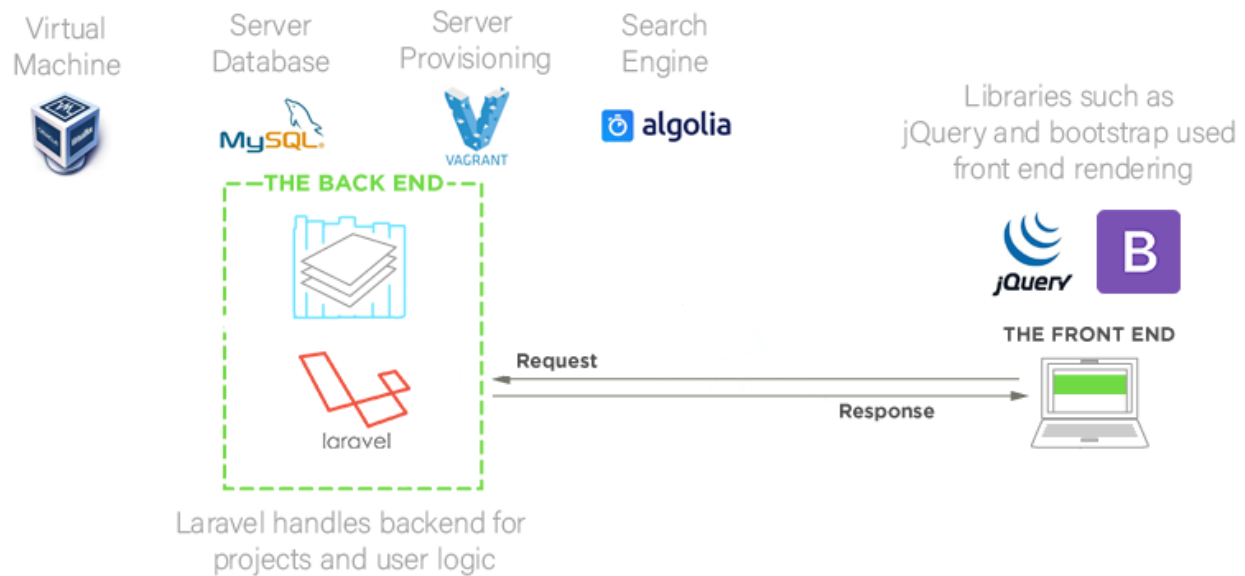


Figure 2: Local technology stack – VirtualBox as the virtual machine, MySQL as the database, Vagrant for building the environment, Laravel for the backend scripting, Algolia for the search engine, Bootstrap and jQuery for the front-end rendering (Wodehouse, 2018)

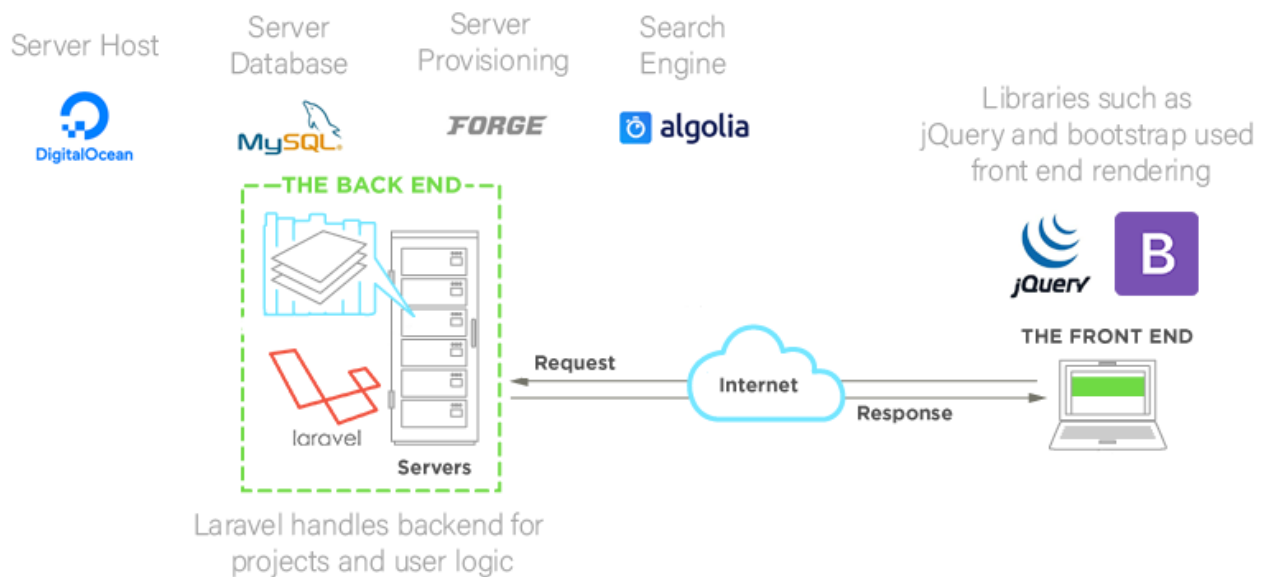


Figure 3: Live technology stack – Digital as the server host, MySQL as the database, Laravel Forge for provisioning the server, Laravel for the backend scripting, Algolia for the search engine, Bootstrap and jQuery for the front-end rendering (Wodehouse, 2018)

3.1.1 Backend: Laravel PHP Framework

A Laravel PHP framework was used for the backend server-side PHP scripting to make the development process faster than if the entire back-end scripting were built from scratch (Otwell, 2019).

Laravel is a PHP web framework which is designed to aid the development of dynamic websites in a standard manner. The framework automates a lot of the infrastructure building that is otherwise required when setting up a website. This includes having libraries for database access, templates as well as setting up user session-management. Having pre-existing templates for different website functionality imbedded in the framework enables optimizing the web development for efficient and clean code (Otwell, 2019). In addition to enhancing the security features of the website, the Laravel framework was also chosen due to the author's previous familiarity with it. This allowed for a faster process of testing and iteration of the website.

Back-end servers require a lot of set-up before being able to run Laravel (Otwell, 2019). These include a range of extensions including the following:

- PHP \geq 7.1.3
- OpenSSL PHP Extension
- PDO PHP Extension
- Mbstring PHP Extension
- Tokenizer PHP Extension
- XML PHP Extension
- Ctype PHP Extension
- JSON PHP Extension
- BCMath PHP Extension

Before deploying the code to a physical server, the author emulated a server environment, locally, on their computer. A virtual box is an emulation of an operating system, in this case Linux, allowing it to have the necessary software extensions installed onto it (e.g. PHP, Nginx, etc.). Oracle VirtualBox (Oracle Corporation, 2019) was used to re-create a server environment on the author's Windows based computer. Laravel Homestead is a pre-configured virtualization package which sets up the virtual box with the right settings to be able to host the Laravel PHP framework (Otwell, 2019). A Vagrant (HashiCorp, 2019) virtual machine, as shown in Figure 4 was used as the middle layer to port the Laravel Homestead package onto the Oracle VirtualBox through the SSH (Secure Shell) network protocol. The SSH protocol, as seen in Figure 5, is a command-line interface that allows Vagrant to securely send information (the Laravel Homestead package in this case) to the Virtual Box.

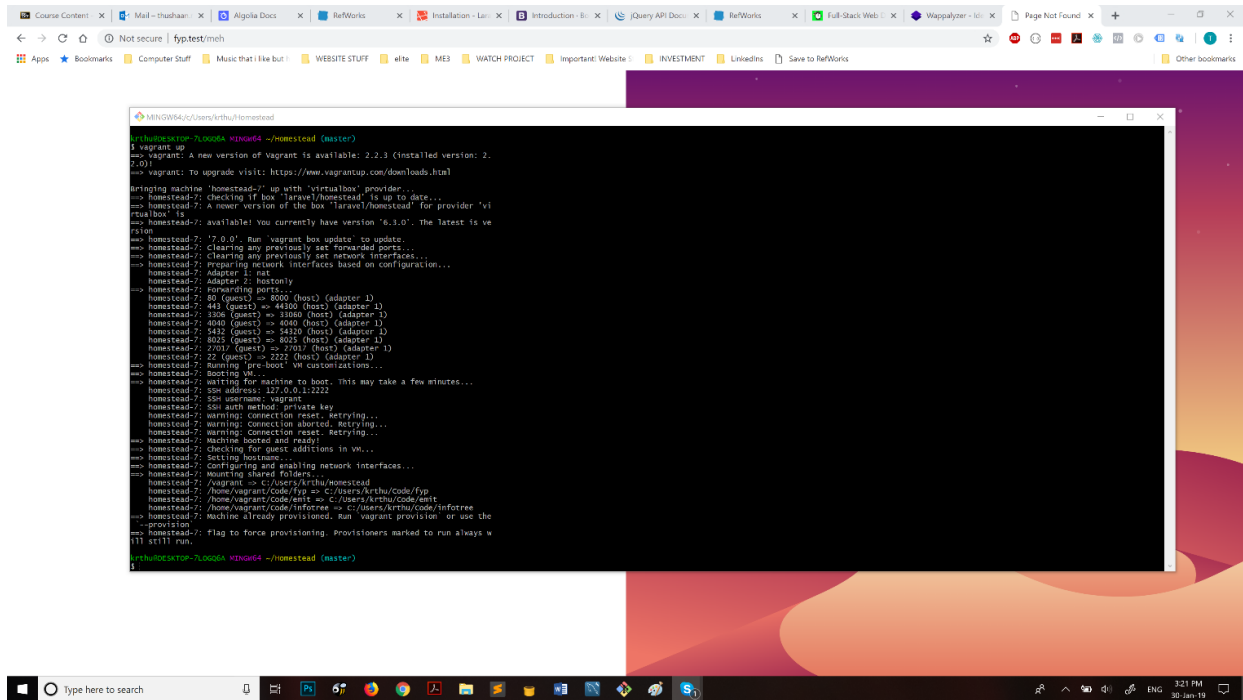


Figure 4: Using vagrant to boot up Laravel Homestead virtual environment

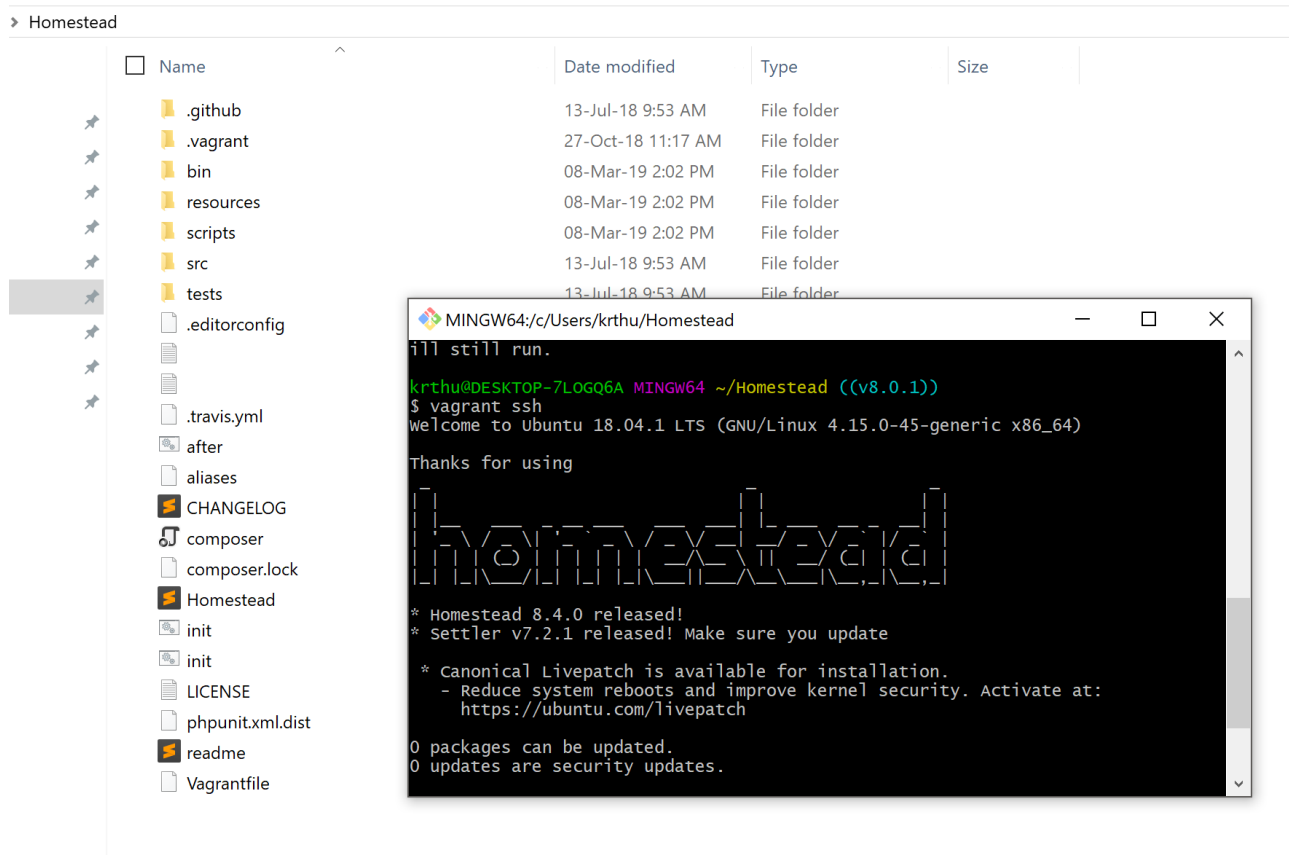


Figure 5: Accessing Laravel Homestead environment through SSH

The website was moved from the local back-end server onto a live server on-line at the end of the project to make it available to other users. At that point, the Laravel framework was hosted on a Digital Ocean server (analogized to the VirtualBox in the local server scenario as shown in Figure 2) through Laravel Forge (analogized to Vagrant).

3.1.2 Frontend: Twitter Bootstrap and jQuery

For the frontend of the website (the code that displays information seen by the browser and thus the user), it was decided to use a combination of the Twitter Bootstrap and jQuery JavaScript frameworks. Both frameworks provide the infrastructure to easily make clean and elegant UX/UI.

Bootstrap is a widely used framework, being the world's most popular front-end framework. It contains CSS- and JavaScript-based design templates for typography, forms, buttons, navigation and other user interface components (Mark Otto, 2019). A principal benefit of Bootstrap is its basic style definitions of "classes" for all HTML elements, which displays them with a uniform appearance (in terms of color, size, font and layout) across various web browsers (i.e. Google Chrome, Apple Safari, Mozilla Firefox and Internet Explorer).

jQuery is a JavaScript library that allows for simple HTML DOM (Document Object Model) tree traversal and manipulation, as well as event handling, CSS animation, and AJAX (Asynchronous JavaScript and XML) (js.foundation, 2019). DOM, as seen in Figure 6, is a logical tree which treats each node within a document as an object of a hierarchy. The jQuery library essentially has standard code which enables the web browser to easily navigate a document, select DOM elements, create animations and even send and retrieve data from a server without interfering with the live display of information on the webpage.

AJAX is a set of web development techniques that enables the browser to send and retrieve information from the server in the background whilst displaying other information to the user (js.foundation, 2019). This primarily allows for a fluid user experience on the front-end, where every new request does not result in a delayed response from the website. The applications of AJAX will be discussed extensively in further sections of the report including Features and Future Work.

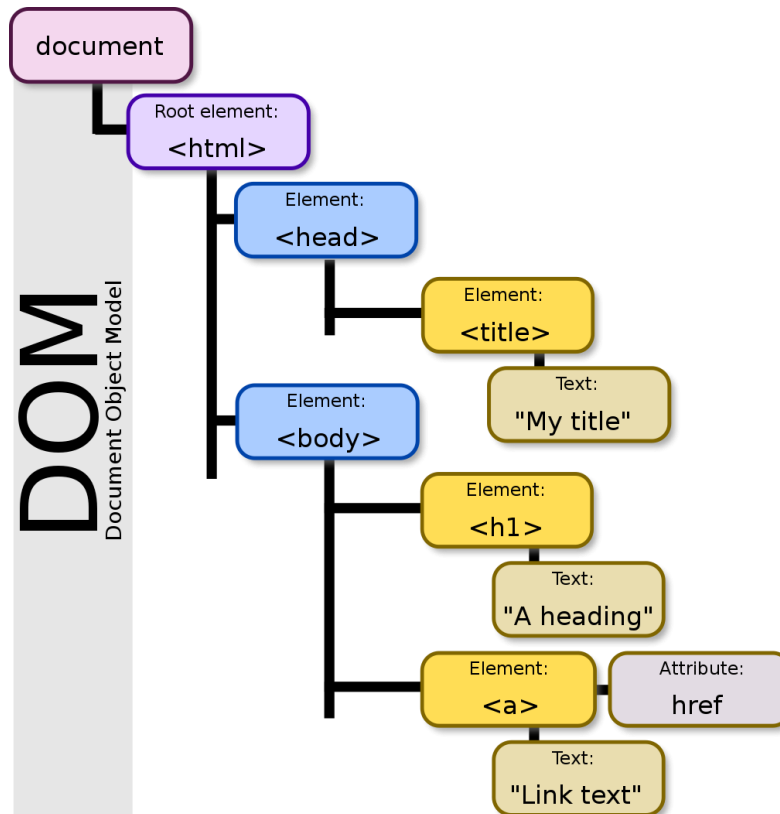


Figure 6: Image showing DOM Hierarchy of a document (Eriksson, 2012)

3.1.3 Repository

All the code for the website was stored in an online repository called GitHub as shown in Figure 7. This stores all the code files and records any changes made to them, therefore acting as a revision control system (Microsoft Corporation, 2016). Github uses a distributed model of storing the code, which means that in addition to storing an online copy of the code, it also constantly synchronizes this onto a local copy on the author's (i.e. website coder's) computer. Using an online repository prevents all the work on the website from being lost and allows for easy integration with Laravel Forge when integrating the live back-end server (Digital Ocean).

Github also can store notes, thus acting as a logbook for all the rationale behind the website's code (Microsoft Corporation, 2016). For future use, the Github repository would also allow easy collaboration when multiple parties are working on updating different parts of the website.

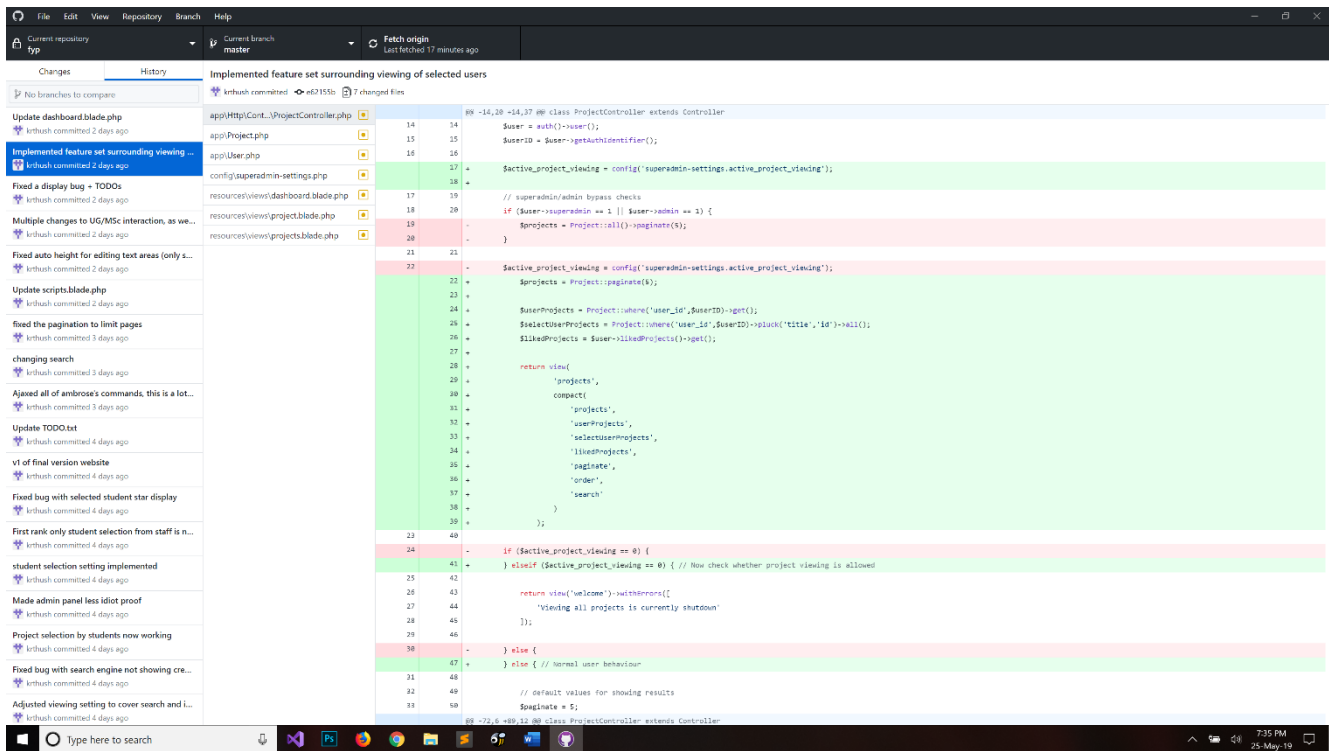


Figure 7: Image showing screenshot of GitHub repository used to store live code

3.1.4 Live Server Hosting

As shown earlier in the live technology stack (Figure 3), the final iteration of the FYP projection website at the end of this project was hosted on a Digital Ocean server provisioned by Laravel Forge (DigitalOcean, 2019). The server comes preloaded with the requirements needed to run the Laravel framework which made deployment of the website easier.

For the purposes of keeping costs low the server was hosted without a domain name, and adjusted viewing was only accessible through sending an IP (Internet Protocol) request. This means that instead of accessing the website through a domain name, an IP address (i.e. a number) specific to the website's server had to be typed into the browser. Due to this it also did not have an SSL (Secure Sockets Layer) certificate and therefore was deemed unsafe. This was acceptable since the website was only live for testing purposes letting other students and staff detect bugs and vulnerabilities.

SSL certificates allow for information shared between the website and users to be encrypted such that it remains private and authentic (Ristić, 2017). A third-party would not be able to intercept the data to either corrupt or eavesdrop on it.

When the live website is implemented for actual use in the following academic year, it will be necessary for Imperial College's IT (Information Technology) staff to incorporate the SAML (Security Assertion Markup Language) system, which will be elaborated in the Authentication section. At that point, it must also be ensured that proper security systems are in set in place, such as having an SSL certificate and making the server capable of handling minor DDOS (Distributed Denial of Service) attacks (Pinto, 2011).

3.2 PROJECT DATABASE

All the information and data used by the website is stored within a database in the back-end. A database is a way of storing large amounts of data in a format which allows for it to be easily used and manipulated. There are two primary types of databases: relational and non-relational. Relational databases such as SQL (Structured Query Language) store data in rows and columns whilst non-relational databases such as NoSQL (Not only SQL) stores data as packets (i.e. each packet contains multiple attributes).

For this project, it was decided that MySQL (relational database) would be the type of database used, as Laravel has inbuilt integration for this (Oracle Corporation, 2019) (Otwell, 2019). The nature of the website's application also demands a relational database since there will be interrelation between the data. For example, a project could be related to a user (staff) and have an associated rank from another user (student). The relation between the different elements of data will be further explored in the Features section.

Object oriented programming (OOP) was the principal coding paradigm used for the website as it allows data to be stored as fields and related to code which is stored as procedures (also known as methods). OOP enables the creation of objects (such as "User"), after which all programming logic can be in relation to this user (object), which in turn can have its own functions. For example, "User->projects()" would be used to show all projects that the user has created.

In Laravel these objects are generally termed as "models". Laravel has a Model View Controller (MVC) php framework, which as shown in Figure 8. In an MVC framework, a given model is the central component (i.e. object) whereas a view is any representation of generated information that the user sees, and a controller accepts inputs from the user and handles all logic behind the generation of views from the models.

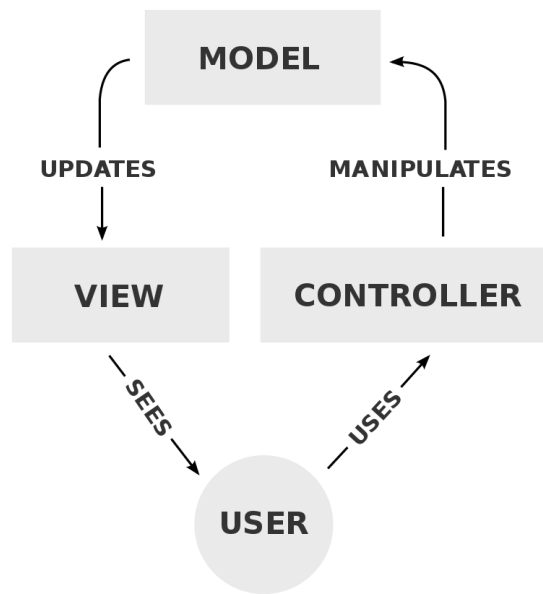


Figure 8: Image showing architecture of Model-View-Controller frameworks (Frey, 2010)

The first stage of the back-end development involved creation of the models, as practiced within object-oriented programming (OOP) (Nixon, 2014). These models behave as the core objects that can be manipulated and stored, examples of which include “projects”, “users”, “likeables” as shown in blue in Figure 9. A specific example of the “project” model is highlighted in red: the model contains key fields such as date of creation, the project’s title, project’s description, which user created the project and categories the project belongs to.

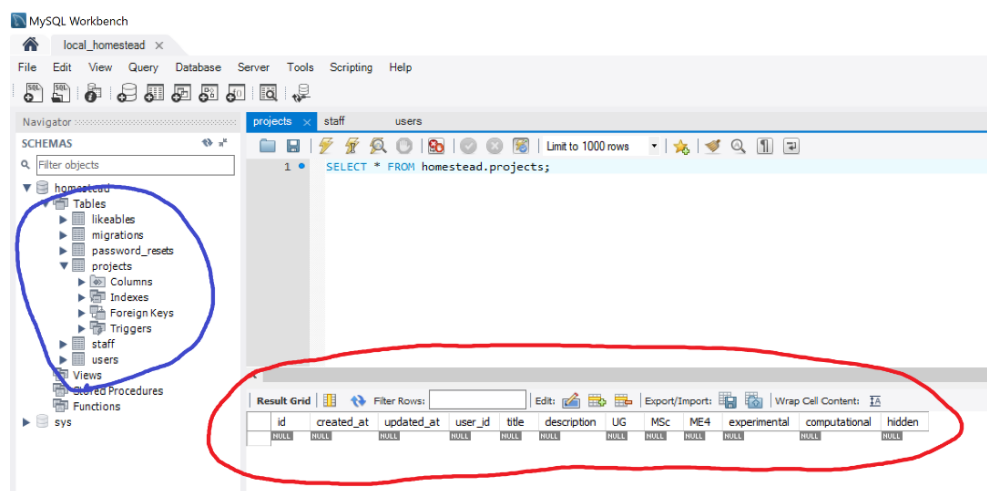


Figure 9: MySQL Workbench showing the website database and its tables in use (Oracle Corporation, 2019)

3.3 SEARCH ENGINE: ALGOLIA

Another major feature that was built into the website was the ability for users to be able search and find projects through a search engine. Although it was possible to build a proprietary search engine, it was decided to use a third-party search engine service as this vastly improved the quality of the search results while still allowing for very intelligent results.

The search engine used in this case was Algolia as shown in Figure 10, which allowed for a great range of configuration. Algolia is a service that offers a search engine to be incorporated into one's website and limit the results to only the website's contents and database instead of also scrapping other parts of the internet, as an engine like Google normally does (Nicolas Dessaigne, 2019). This also leads to faster and more accurate search results.

The search engine index ranks each "project" according to what the user searches. The results are presented as a list of objects with each project being an object with a range of attributes that are used to determine its rank.

This meant that as a default setting, each search request resulted in the website to search through all fields of a project (including the description, authors, dates, etc. of a project) and rank them based on a variety of relevance indicators. This makes sure that even if the user searches for a string (i.e. text) that isn't exactly found in a title, the engine looks for the string in the other fields, which contribute to the projects' ranks.

The search engine was also used to create replica indices (which are cloned lists of the projects) but with different edited configuration allowing for actions such as ranking the projects by alphabetical order, date of being created, and other such instances. This meant that it was possible to allow the user to order the results of their project search in a manner most suitable for them, as will be detailed in the following Features Section 4 when looking at the search feature.

In addition to accounting for typing mistakes that a user makes, Algolia can also be used in the future for learning from user searches and predicting the types of results expected according to specific things that are looked for, further discussed in Section 6.3.2.

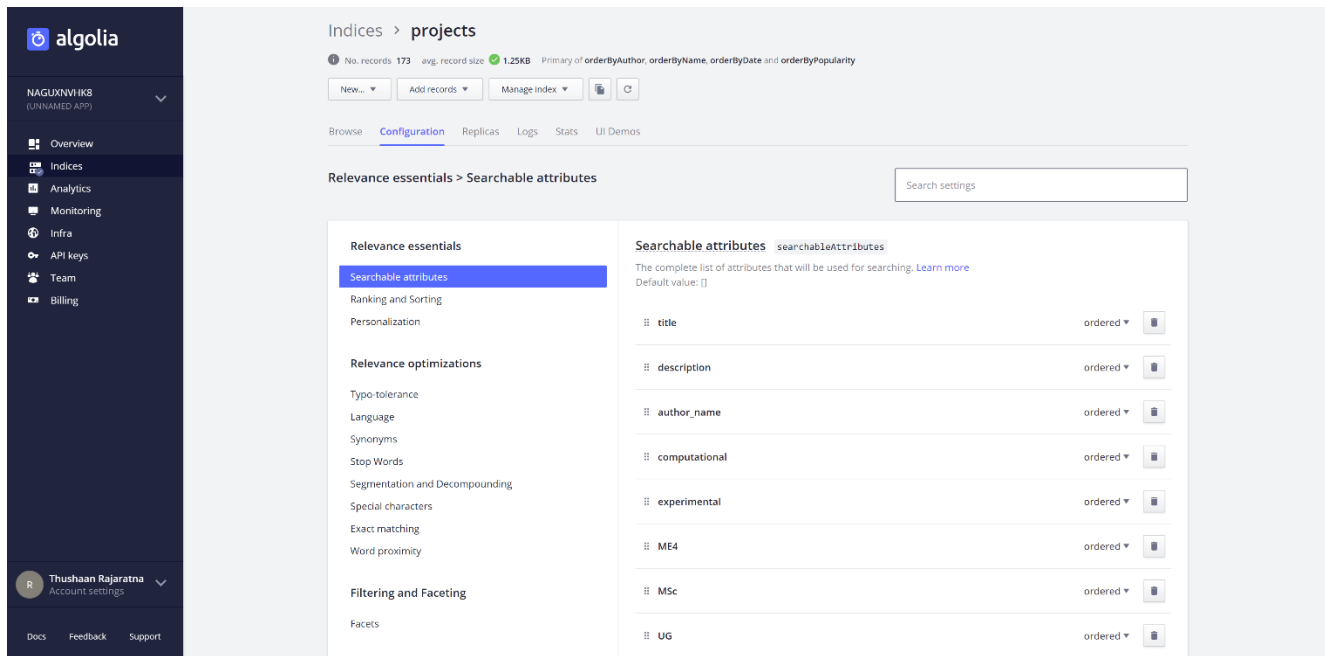


Figure 10: Back-end of "Algolia" search engine in use

3.4 SECURITY

Information security was an important aspect of the website since it will be handling sensitive information such as user passwords and university IP (Intellectual Property) protected details about the final year projects. In order to ensure that only appropriately authorized users have access to relevant information and to prevent malicious hacks of the information, several tactics were incorporated as outlined below.

The back-end php framework used, Laravel, already has built-in security features that have now become the norm by web development standards (Otwell, 2019). This provided a starting platform off which to build additional functions to improve the website's security.

One pre-existing feature within Laravel's framework is authentication, which uses "guards" and "providers" to ensure the right user is logged in at any one time with the right permissions (Otwell, 2019). This ensures that each user is granted access to only those parts of the website that be-fit them. For example, students would not be able to add or edit projects and their details. This system also prevents anyone apart from registered students or staff from accessing the website.

Another security feature within Laravel is Bcrypt, which is a third-party password hashing function. Bcrypt cryptographically stores user password details, where instead of storing their passwords as plain text it stores each password as a pseudo-randomly allocated value with a corresponding key. The key is randomly allocated by an algorithm. This means that if the database of user passwords is ever leaked, it would not reveal the actual characters used by users but instead only contain keys, which would be close to meaningless without the attacker knowing the algorithm used to generate those keys. This prevents leakage of user passwords and ensures that if there is an attack on the website – users' passwords from their accounts on other services will not be directly compromised.

Three common types of attacks on a website are: CSRF (Cross-Site Request Forgery), XSS (Cross-Site Scripting) and SQL injections (Shema, 2012).

CSRF occurs when a third-party, other than a user attempts to “session-ride” on a user's interaction with the website (Shema, 2012). i.e. once the user logs into their account, a malicious third-party tries to make use of the open gateway of interaction between the user and the server to also access the website. Laravel uses CSRF tokens to ensure that a user logging into their account is not ‘free-rid’ on by a third-party (Otwell, 2019). At the beginning of a user's session, Laravel allocates the user a unique token which it uses to constantly check that data packages are only sent and decryptable by the user's specific computer. This means that even if a third-party attempted to ‘hack’ the user's interaction with the website, they would not be able to view any of the information.

XSS attacks are a malicious third-party's attempt to inject code into the website, which will then be sent to an unaware user accessing the website and potentially corrupt their computers (Shema, 2012). Laravel can be used to properly validate and sanitize data before it is inputted and stored onto the website (Otwell, 2019). This means that Laravel checks for the type of data being input, and only accepts and stores it onto the website if it is of a valid format (numerical, date, currency, text, etc.) as pre-defined by the website creator. Laravel encourages the user to escape all outputs by using the “{{ }}” character in Blade, which is an in-built tool incorporated to make preventing XSS attacks much easier.

SQL injection attacks are when malicious code is attempted to be input into the back-end script of the website server to tamper with the stored information (Shema, 2012). This means that unauthorized third-parties could change the information about students, staff and even

projects and corrupt the data. Laravel uses a technique known as PDO (PHP Database Objects) bindings to prevent SQL Injection attacks, which ensures that each function has a parameter value bound to it, which is executed separately with a different protocol (Otwell, 2019). A third-party's SQL code would not work since they would not have the right protocol to execute the code parameter.

As mentioned earlier in Section 3.1.4, obtaining an SSL certificate for the website in the future would prevent unauthorized users from snooping on sensitive data as it cryptographically encodes the data (Garfinkel, 2002).

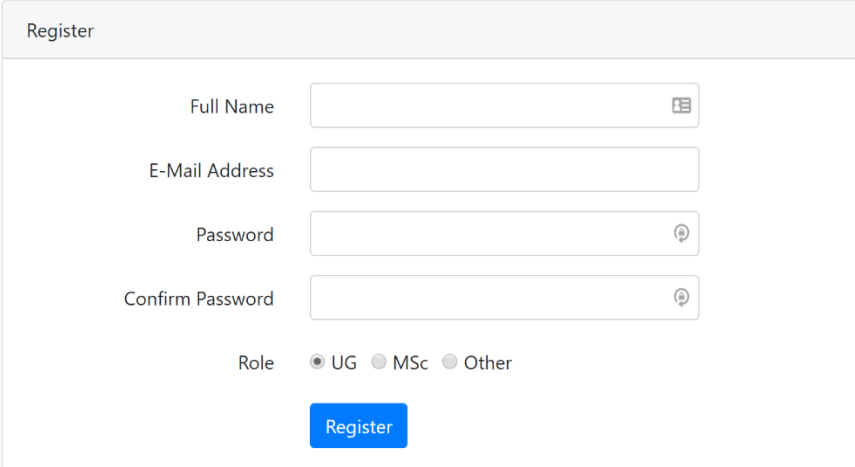
As discussed further in Section 5.3, a student even tried to implement some of these low-level hacking attempts (SQL injection, XSS attacks, etc.) and failed in doing so, indicating that it is unlikely that a student will be able to easily hack the website.

4 PROJECT ALLOCATION WEBSITE – FEATURES

4.1 AUTHENTICATION

The first step in implementing the FYP project allocation website is to create the “User” model since all following models shall be most linked to this model in some shape or form. This means that every action on the website begins with identifying what user is accessing the website, and subsequently detecting what permissions they have and what information they can see.

The Laravel framework comes with a pre-packaged authentication scaffolding (called Auth) which made the development process of a user authentication system faster. In conjunction with Laravel php artisan, the command line interface used in Laravel, it was relatively straight forward to create all the necessary databases, models, controllers and other building blocks to set-up a list of users. As seen below, in Figure 11, the authentication system enabled the user to register initially and then access the website by inputting certain log-in details.

A registration form titled "Register" with a light gray header. The form contains four input fields: "Full Name" with a calendar icon, "E-Mail Address", "Password" with an eye icon, and "Confirm Password" with an eye icon. Below these fields is a "Role" section with three radio buttons: "UG" (selected), "MSc", and "Other". At the bottom is a blue "Register" button.

Register

Full Name

E-Mail Address

Password

Confirm Password

Role ☒ UG ☐ MSc ☐ Other

[Register](#)

Figure 11: Registration view for a guest user

4.1.1 Middleware

Middleware acts as a bridge between a request (for information from the back-end server) and a response (to the front-end server). It is a type of filtering mechanism. It determines after receiving a request, if the user has the correct permissions, and accordingly decides what response to direct back to the browser (front-end of the website). The Auth scaffolding itself creates the authentication middleware needed to ensure that only logged in users can access relevant parts of the website. It is a Global Middleware, in that it tests for this authentication at every request (not just specific ones) across the website.

Shown in Figure 12 are all the web routes are handled by the website. In Laravel, routes act as an area (highway) for managing all the different request options a user can make and deciding what 'pathway' of functions will be performed for the user to receive a response.

```

RegisterController.php x web.php x LoginController.php x
<?php

/*
-----
Web Routes
-----

Here is where you can register web routes for your application. These
routes are loaded by the RouteServiceProvider within a group which
contains the "web" middleware group. Now create something great!
*/

Route::get('/', function () {
    return view('welcome');
});

Route::get('/home', function () {
    return view('welcome');
});

Auth::routes();

// Use the built in auth middleware to allow only logged in users
Route::group(['middleware' => ['auth']], function() {
    // Routes that need auth:

    // project routes
    Route::get('/dashboard', 'ProjectController@dashboard')->name('dashboard');
    Route::get('/projects', 'ProjectController@projects')->name('projects');
    Route::get('/projects/{project}', 'ProjectController@show')->name('project');
    Route::post('/projects/new', 'ProjectController@store')->name('new-project');
    Route::delete('/projects/delete', 'ProjectController@destroy')->name('delete-project');
    Route::patch('/projects/update/{project}', 'ProjectController@update')->name('update-project');
    Route::get('projects/match/{project}/{student_id}', 'ProjectController@match')->name('match-project');
    Route::get('projects/unmatch/{project}/{student_id}', 'ProjectController@unmatch')->name('unmatch-project');

    // like routes
    Route::get('projects/like/{project}', 'LikeController@like')->name('like-project');
    Route::get('projects/rankup/{project}', 'LikeController@rankup')->name('rankup-project');
    Route::get('projects/rankdown/{project}', 'LikeController@rankdown')->name('rankdown-project');
    Route::patch('projects/reorder', 'LikeController@reorder')->name('reorder-projects');

    // search routes
    Route::get('/search', 'ProjectController@search')->name('search-projects');

    // super admin routes
    Route::get('/superadmin', 'SuperAdminController@show')->name('superadmin');
    Route::get('/superadmin/export-users', 'SuperAdminController@exportUsers')->name('export-users');
    Route::get('/superadmin/export-projects', 'SuperAdminController@exportProjects')->name('export-projects');
    Route::get('/superadmin/export-selected-project-users', 'SuperAdminController@exportSelectedProjectUsers')->name('export-selected-project-users');
    Route::get('/superadmin/toggle-project-viewing', 'SuperAdminController@toggleProjectViewing')->name('toggle-project-viewing');
    Route::get('/superadmin/toggle-project-selection', 'SuperAdminController@toggleProjectSelection')->name('toggle-project-selection');
    Route::get('/superadmin/toggle-project-first-matching', 'SuperAdminController@toggleProjectFirstMatching')->name('toggle-project-first-matching');
    Route::get('/superadmin/toggle-project-all-matching', 'SuperAdminController@toggleProjectAllMatching')->name('toggle-project-all-matching');
});

```

Figure 12: PHP code from routes/web.php file, showing all web requests Laravel manages for this website

The routes also show an origin of the basic logic behind the website. An example of such logic is shown in Figure 13, in the specific case of user registration/login. It must be noted that there is logic applied to all aspects of the website.

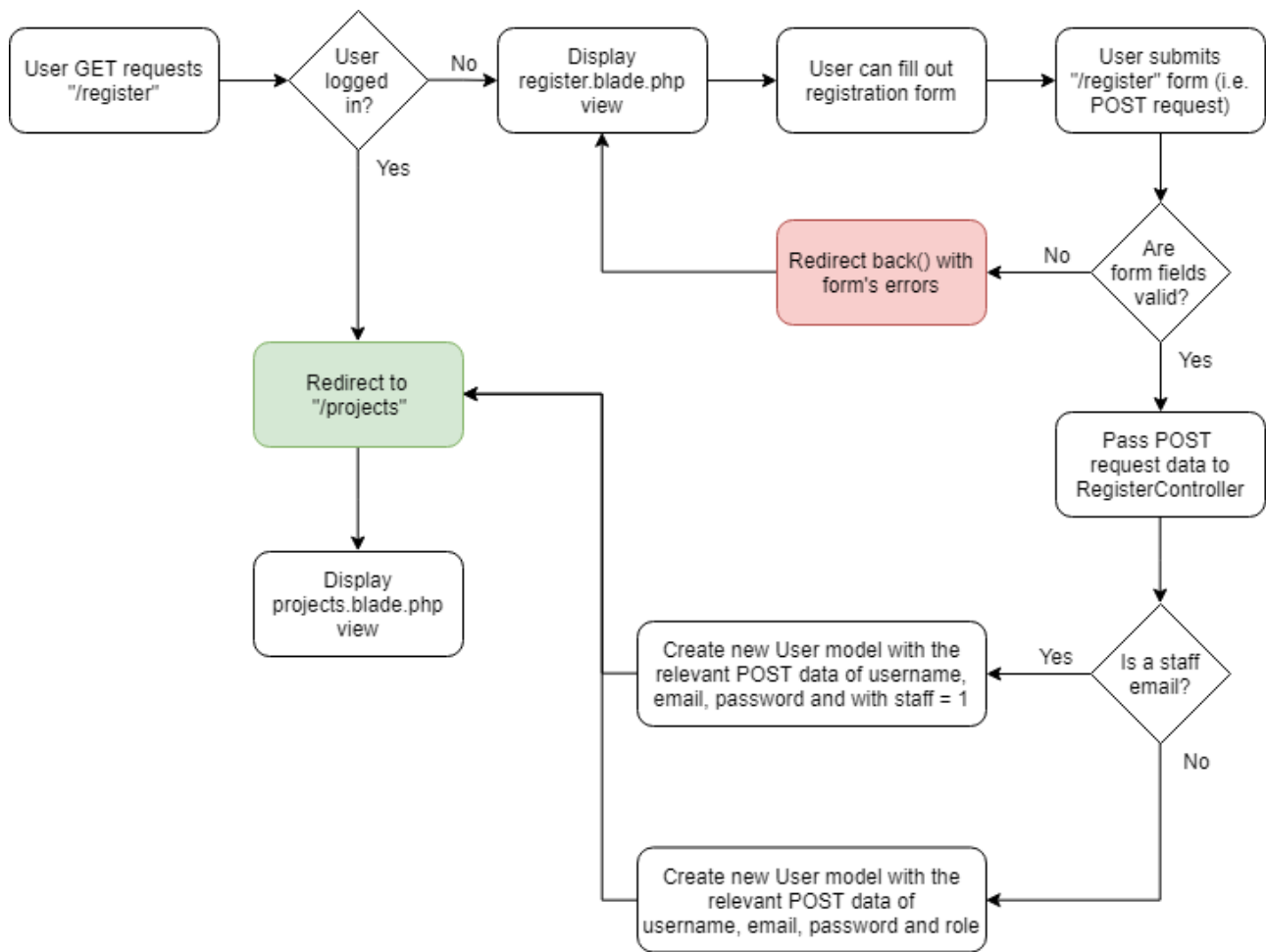


Figure 13: Diagram showing general logic behind code handling user registration

4.1.2 User Roles

As noted in Figure 11, the website allowed for creation of user roles, which was conveyed as an additional attribute in the user's database (e.g. Boolean values of true/false for staff, UG, MSc, admin, superadmin). Students could choose to be UG (Undergraduate) or MSc (Master of Science). This essentially categorized a user into staff, students or administrator of the website, each having different levels of permissions to modify the website. Whenever other components of the website were loaded, checks were done on the user's role as to whether they should be viewing the relevant information, and then also whether they should be allowed to make such actions. The logic behind some of roles even involved manipulation of the website to extreme levels, such as a "super admin" being able to access as well as edit all parts of the website and export important information from it.

The website was designed such that a user could only choose to be "UG", "MSc" or "other". By choosing the "other" option, the logic shown in Figure 13 will check existing staff

databases to see if the registered email is meant to have a certain higher class role, which is predetermined by the final year project coordinator (Ambrose Taylor). For increased convenience to the coordinator, a future feature for the website could be to add these users of higher permission directly from the “superadmin” interface panel.

Having established the different types of user roles, the remainder of the website logic was built where every other the other model was dependent on the user model.

4.2 CREATING, DELETING AND UPDATING PROJECTS

A necessary feature for the website was for users registered as “staff” to be able to create a “project”, which would allow them to add onto the list of existing final year projects, their own project and its descriptions. “Project” was then chosen as another model that was next created. The “project” object will serve as the fundamental building block which all created projects will mimic. This includes manipulating the object to do more complex functions such as search, selection, ranking and so forth.

The website logic was such that a “project” would belong to a “user” and that “users” can have many “projects”. This logic was handled in the declarations of the “project” and “user” object by routing methods as discussed earlier in Section 4.1.1.

Each “project” model had fields such as “title”, “description”, “user_id” (this would show who made the project) and so forth which distinguished it from the rest. Upon discussion with the supervisor, the number of fields were expanded to included “UG”, “MSc”, “Hidden”, etc. The increase in number of fields led to the generation of more features such as only UG projects being shown to UG students, and similarly MSc projects only to MSc students. “Hidden” projects would not be visible to students, only visible to staff who created them, admins and the superadmin. Figure 14 shows a member of staff’s view of an individual project they have created and can edit. The dashboard view relating to student selection in Section 4.5, allows for creating and deleting projects. It also displays all the viewable fields for a project.

In general, the benefit of using such a basic “project” model building block is that it is very easy to modify the object in the future if more features are needed. This means that when the website is handed over (as discussed further in Section 6.2), future admins and developers can very easily add more fields (such as allowing a project to have “co-supervisors” for example) and modify the website to handle such fields for features.

D,M,T a new artificial patch for repairing localised damage to the surface of a human shoulder joint

Supervisor: Amis

(★ Student Allocated ; student)

Although there are existing products for total replacement of a damaged shoulder joint, this project aims at an earlier stage of the process, when the damage is still localised and follows a dislocation injury. The plan is to design, make and test a new method using an artificial patch fixed to the bone within the head of the humerus. The prototypes will be made by additive manufacturing in a rapid prototyping machine. The shape of the surface will have to match the natural shoulder, obtained from medical scans. In addition to getting a surgeon to implant the patch into plastic replica bones, it may then move on to testing in a dead knee. Aspects such as designing the patch, the instruments required by the surgeon, and then measuring the accuracy of restoring the surface geometry of the shoulder will arise as the work progresses.

☒ Suitable for UG
☐ Experimental

☒ Suitable for MSc
☐ Computational

☐ Hidden

Delete Project

Edit Project

Number of students having selected this project: 2
Students who have selected this project and their rank:

student	1	Deselect Student
thush	1	Select 2nd Student

Figure 14: Individual project page view as a staff user - shows all attributes projects have

4.3 PROJECT VIEWING

Note that projects can be searched through a very advanced search engine, built on Algolia, as mentioned in Section 3.3. The results can be ordered by many different attributes such project name, author name, alphabetically, creation date and project popularity (i.e. how many users have selected it so far). The number of results displayed can be varied from 5 to showing all results. The page displaying these results, shown in Figure 15, by default is the homepage for most users.

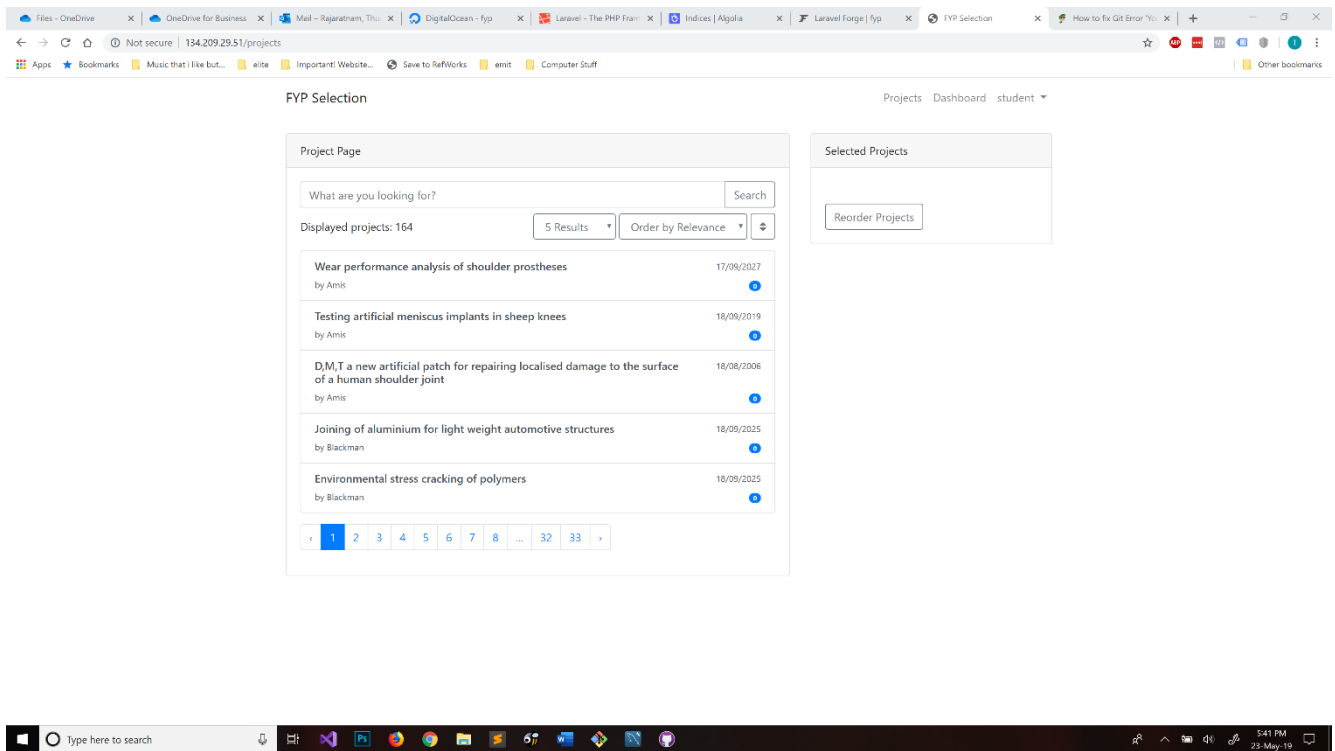


Figure 15: Screenshot of "projects" homepage showing all projects, the search bar and any selected projects

There was a focus on making the website design (UX/UI) very easy to navigate, for which a component known as “modals” from Bootstrap was heavily used. This method of seeing projects is shown in Figure 16. This meant that when a user views a project, rather than having to navigate away to another new page, the project would “pop up” and overlay over the current window, allowing the user to read about it and select it if he or she wishes.

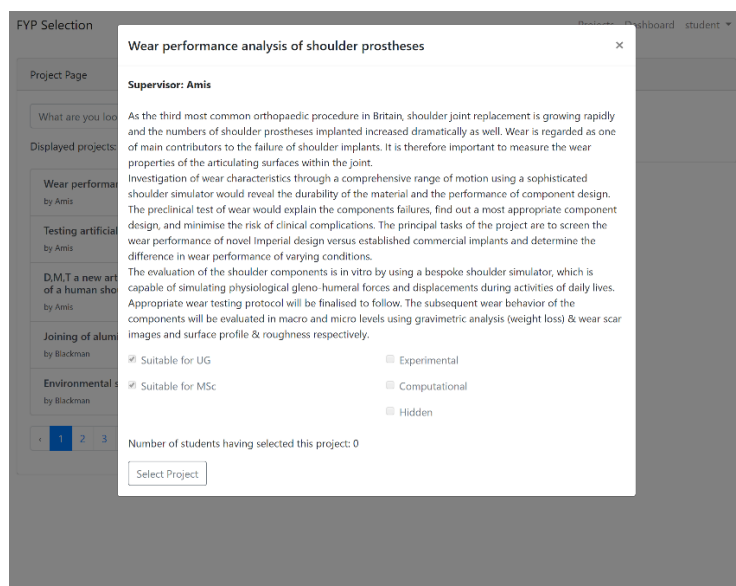


Figure 16: Individual project shown as a "modal"

4.4 PROJECT SELECTION AND RANKING

The method of selecting and ranking projects was also made much easier within the website's user interface. As discussed in Section 4.4, users can easily select a project while in the same viewing panel (although it should be noted that as soon as the project is selected the page is refreshed). This action can be made more fluid by using AJAX as discussed in Section 6.3.1.

Once the projects are selected the users can easily rank up and rank down the projects through dragging and dropping them after clicking the “re-order projects” button. This is shown in Figure 17. This makes the UX/UI far easier and more fluid, which is also enabled by AJAX which allows for the front-end to display certain information without refreshing the page whilst the back-end processes the next set of required information.

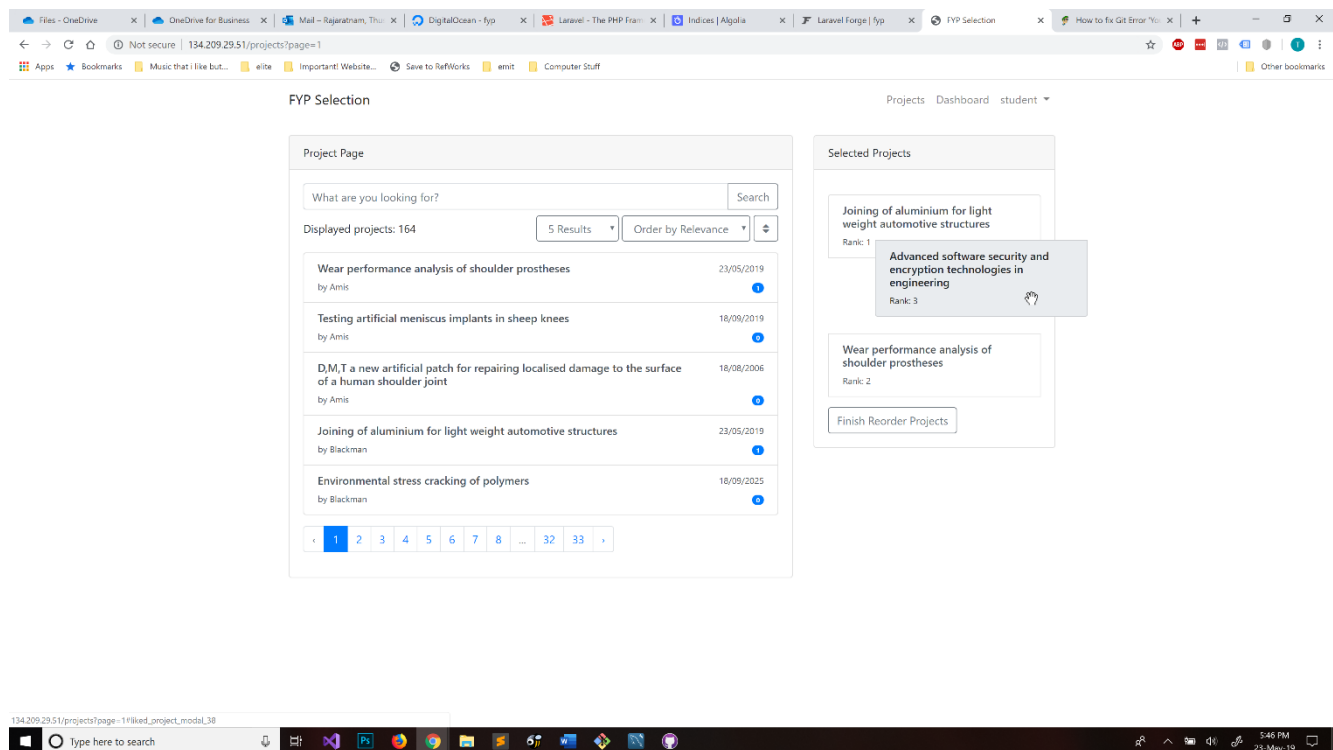


Figure 17: Screenshot showing easy drag and drop method of ranking projects

4.5 STUDENT SELECTION AND PROJECT MATCHING BY STAFF

The dashboard for the staff is a simple table view of all the projects they have created and all the students who have selected their projects as one of their top 3 choices. The layout of this dashboard was heavily influenced by the project supervisor due to his extensive experience of managing the final year projects program and interacting with a broad range of staff. The

dashboard (shown in Figure 18) allows for staff to quickly understand what projects they have created and also shows them which students they must analyze to select for their projects. They can then select students for a project by clicking on the individual project and then selecting the student from that project's page as discussed in Section 4.2. It should be noted that staff also have the option to select 2 students for a single project – this feature was added on the request of project coordinator.

FYP Selection Projects Dashboard Amis ▾

Dashboard Delete Project Add Project

Below are the projects you have created: ★ Selected Students

Project Name	1st Choice Students	2nd Choice Students	3rd Choice Students	Last Updated At
Wear performance analysis of shoulder prostheses		George Adams		23/05/2019
Testing artificial meniscus implants in sheep knees				18/09/2019
D,M,T a new artificial patch for repairing localised damage to the surface of a human shoulder joint	student ★ thush			23/05/2019

Figure 18: Dashboard showing overview of created projects with which respective students have selected them and their preference rank

It is worth noting that as mentioned earlier in Section 4.2, staff can also create and delete projects from this panel. The method in which they do this, uses modals like Section 4.3 as when students view projects, this again allows for a very simple and fluid UX/UI. This is shown in Figure 19.

FYP Selection Projects Dashboard Amis ▾

Dashboard Delete Project Add Project

Below are the projects you have created: ★ Selected Students

Project Name	1st Choice Students	2nd Choice Students	3rd Choice Students	Last Updated At
Wear performance analysis of shoulder prostheses		George Adams		23/05/2019
Testing artificial meniscus implants in sheep knees				18/09/2019
D,M,T a new artificial patch for repairing localised damage to the surface of a human shoulder joint	student ★ thush			23/05/2019

Add Project ✕

Title

Description

☒ Suitable for UG
☐ Suitable for MSc
☒ Experimental
☐ Computational
☐ Hidden

Submit

Figure 19: Project creation/editing using "modals"

4.6 SUPER ADMIN PANEL

The superadmin panel which can only be accessed by the project coordinator, gives access to site-wide configurations and the ability to extract useful information about staff and student decisions. In the database, the coordinator was given an attribute of superadmin as “true”. Similarly, there is a Boolean value of admin (true/false) attributes in the user database, which enables any user categorized as “admin” to have extra privileges such as viewing all projects and editing them. This categorization of users as “admins” also enables future expansion of functions, such as the super admin panel being modified to allow for selection of admins.

As shown in Figure 20, the superadmin panel enables the project coordinator to control the level of features available to other users, including project viewing and selection for students and the ability for staff to allocate students to their projects. If needed, the coordinator could also allow the staff to only be able to choose the students who’ve selected a project as their first rank. This allows for phased rolling out of the selection process, i.e. allowing for only 1st rank students to be matched first to a project and then unmatched students to be given a second chance for looking at projects.

These settings were coded using a Laravel package called Laravel-Config-Writer, which enabled the super admin to customize different information viewing options for other users. Another Laravel package used was Laravel-Excel (Maatwebsite, 2019), which enabled the website to export different lists (students, projects, etc.) as Microsoft Excel spreadsheets which was deemed useful for the project coordinator.

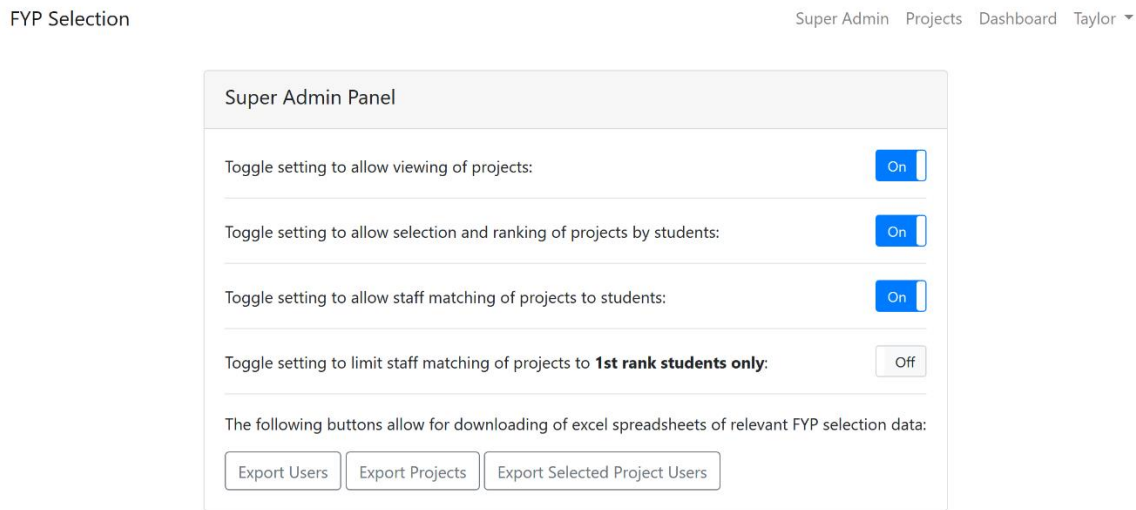


Figure 20: Super admin panel for Ambrose to configure site wide settings

Shown below in Figure 21 is an example export of all projects, where all the attributes of a project are shown (e.g. its title, description, its author's id, etc.). It should be strongly emphasized that even though these are attributes that are stored in the MySQL database mentioned in Section 3.2, the exported attributes can be easily modified to extract a great number of more fields (e.g. names, emails, CIDs etc.) The expanded list of attributes is left to be determined during the integration of the website with Imperial College's system, further discussed in Section 6.2.2. At that point it shall be better known what the minimum number of attributes needed for both the university and the project coordinator to manage the FYP allocation process are.

id	created_at	updated_at	user_id	title	description	UG	MSc	experiment	computatic	hidden	popularity	selected_u	selected_user2_id	
1	2027-09-17	2019-05-27		1 Wear perc	As the thirc		1	1	0	0	0	1	0	0
2	2019-09-18	2019-09-18		1 Testing arti	The medial		0	1	0	0	0	0	0	0
3	2006-08-18	2006-08-18		1 D,M,T a ne	Although tl		1	1	0	0	0	0	0	0
4	2025-09-18	2019-05-25		2 Joining of a	Component		1	1	1	0	0	1	0	0
5	2025-09-18	2025-09-18		2 Environme	Under the i		1	0	1	0	0	0	0	0
6	2025-09-18	2025-09-18		2 Analysis of	Adhesive jc		1	1	1	1	0	0	55	0
7	2025-09-18	2025-09-18		2 An investig	Modern inc		1	0	1	0	0	0	0	0
8	2026-09-18	2026-09-18		3 Scrutinizing	The therm		1	1	0	1	0	0	0	0
9	2012-10-18	2012-10-18		3 Developing	X-ray driver		1	1	0	0	0	0	0	0
10	2027-09-17	2027-09-17		4 Tribology o	Food senso		1	0	0	0	0	0	0	0
11	2020-09-18	2020-09-18		5 Reliable co	The Non-Dr		1	1	0	1	0	0	0	0
12	2024-09-18	2024-09-18		6 Acoustic le	This project		1	1	1	0	0	0	0	0
13	2014-08-18	2014-08-18		6 Acoustic co	Convention		1	1	0	0	0	0	0	0
14	2014-08-18	2014-08-18		6 Electro-ma	This project		1	1	0	0	0	0	54	0
15	2025-09-18	2025-09-18		7 Rheologica	Caramels a		1	1	1	0	0	0	0	0
16	2025-09-18	2025-09-18		7 Investigatir	Panel paint		1	1	1	1	0	0	0	0
17	2025-09-18	2025-09-18		7 Rheologica	Aerated chi		1	1	1	0	0	0	60	0

Figure 21: Example download of spreadsheet showing all projects

5 TESTING

Once the website was completed, its features and usefulness were measured through testing. The testing involved testing the website's ability to store data, resist security threats and compatibility with a range of different devices that would be used to access it. Furthermore, feedback from all the different stakeholders using the website was also obtained to iteratively make changes.

5.1 SAMPLE DATA

During the rapid prototyping of the website it was important to make sure that the website was regularly tested to ensure everything was functioning as intended.

The most important test that was conducted was inputting a batch of sample data into the website. The sample data was created using the output of the 2018 selected FYPs from the existing project allocation website as shown in Figure 22. The creation of both, the user models as well as the project models were formed by manipulating and editing the original data. This showed that the website was successfully capable of storing and loading all the relevant models.

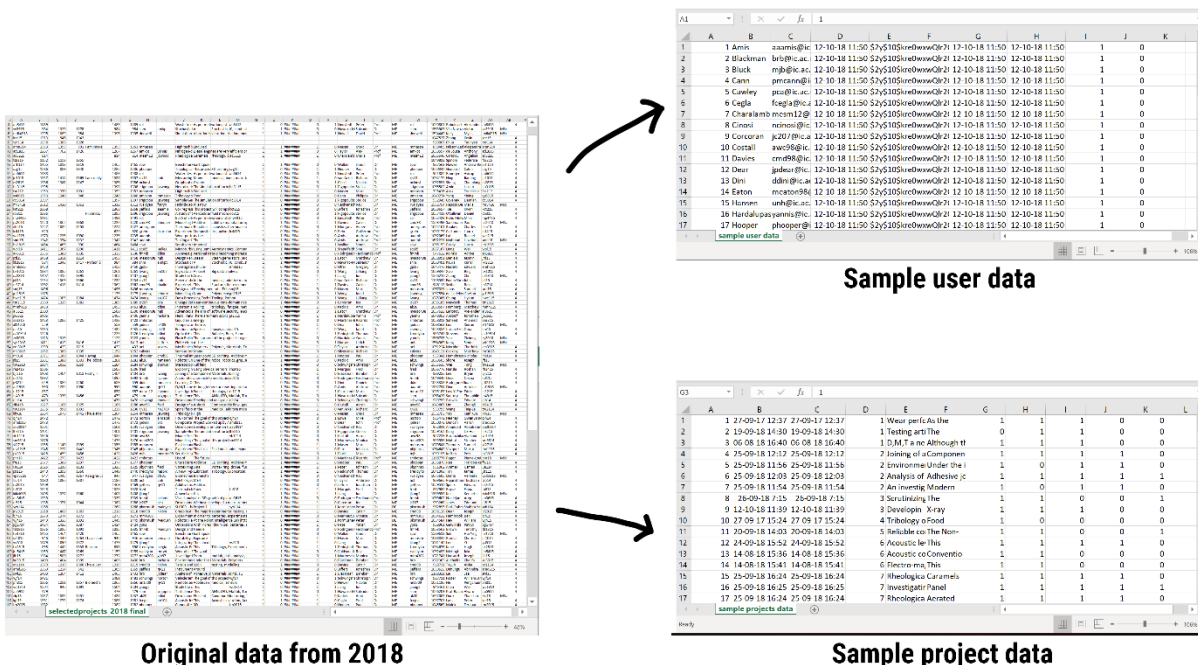


Figure 22: Creation of sample data from 2018 FYP data

5.2 USER FEEDBACK

5.2.1 Student Feedback

Showing the website to current students enabled the author to obtain suggestions of features of the website that they would like to improve their user experience.

The types of features requested by students included the ability to search through the list of projects, filter them by different categories (including whether a project was experimental or computational in nature) and even have them associated to certain tags and keywords. This inspired the use of Algolia to build a search engine within the website that allowed for searching. Students reported to be satisfied with the search function and believed it improved

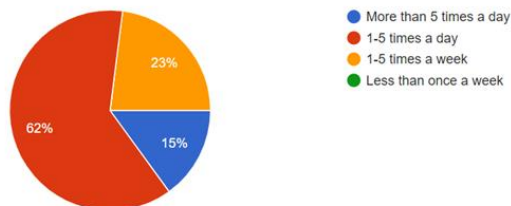
their experience of selecting projects through the new website. There is now room to improve the search feature with the ability to search for projects tagged with certain words.

Testing also revealed that students would prefer the ability to view images regarding the projects on the website, as well as be suggested similar projects to the one they are viewing at any given time to aid their selection process. Students also mentioned that the login system could be improved such that it allows them to directly use their Imperial College login details without the need to register an additional profile. These improvements are further discussed in Sections 6.3.2, 6.3.3 and 6.3.4.

Surveys were also done with regards to students' feelings towards their use of existing university software. Although, it should be noted that results from such surveys are not very reliable, especially since the users are using a system over which they have very little choice – since it is the only option they have provided from the university. The survey was done about Blackboard as it was difficult to run tests on users who used the previous FYP system, as they were all final year students who have mostly graduated and left the university. Shown below in Figure 23 are the results for this test.

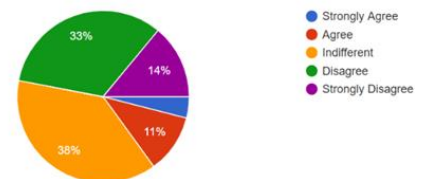
How often do you use blackboard?

100 responses



Do you find discussion on blackboard satisfactory?

100 responses



Do you find the organising of course page structures on blackboard satisfactory?

100 responses

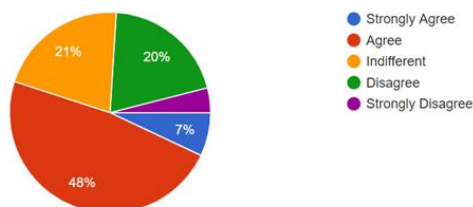


Figure 23: Survey on student's VLE usage (i.e. blackboard)

It is quite hard to see a general trend, although students are unsatisfied when it comes to discussion of topics on Blackboard. This is something that can be improved upon on the FYP selection website and is something that could be a future improvement – specifically implementing more chat and communication-based features into the website.

Again, it is worth highlighting the ineffectiveness of surveys, especially when it comes to software. User testing and focus groups created based on that testing would be far more valuable. Unfortunately, due to the lack of resources, such as the inability to convince students to use a website which has no utility to them as of now (i.e. it is only useful for them when they are selecting projects, and by that point it will be too late to run such tests), it was not possible to conduct more accurate user tests.

5.2.2 Staff Feedback

The main portion of the staff feedback was taken from Ambrose Taylor (who is not only the supervisor for the project, but a staff member who would use the website for selecting projects and would be the superadmin who would manage the entire FYP selection procedure). Based on his feedback many features relating to the staff and admin parts of the website were created as well as modified for improvement.

Features such as the superadmin panel (see Section 4.6), the staff dashboard (see Section 4.5) and the project panel (see Section 4.2), were all made with his guidance and advice.

5.2.3 Iteration based on Feedback

The continuous feedback obtained from both the staff and students were used to iterate different features and designs of the website as it was being created to provide a more satisfactory experience to its stakeholders.

Shown in Figure 24 below is a very early version of the search engine which was improved based on obtained feedback to incorporate features such as modal views, pagination options and more rigorous ordering options. This made the search function more intuitive for users to view results from, as well as more thorough in being able to sort projects.

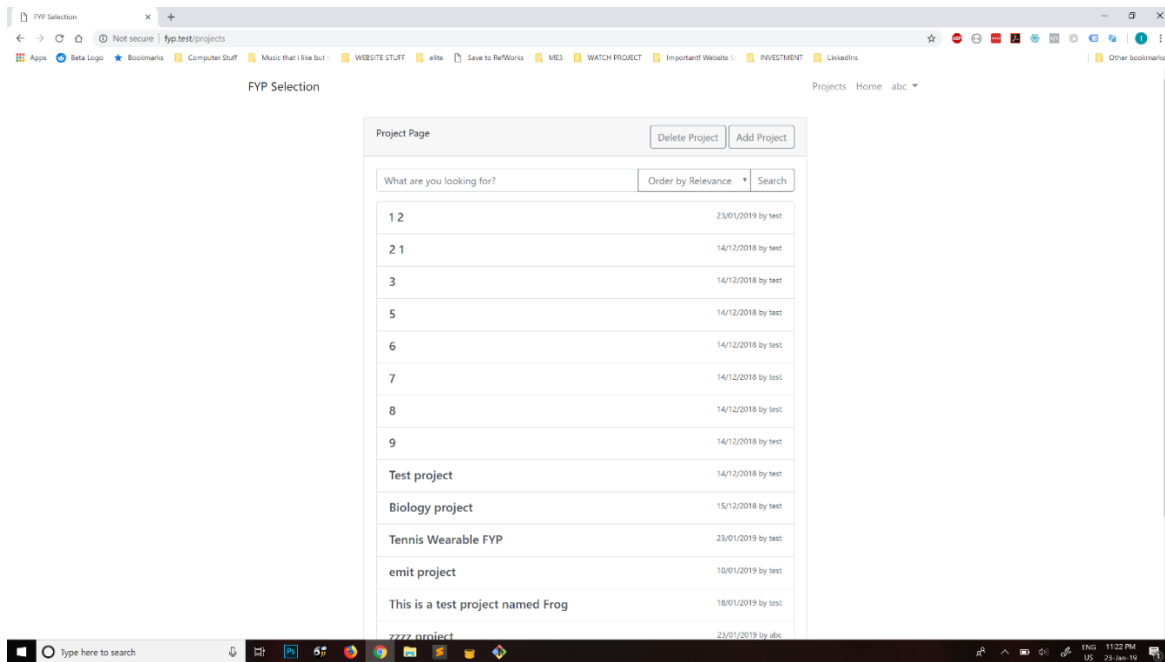


Figure 24: Screenshot of an early version of the search (i.e. projects) page in January

Shown in Figure 25 below is also is a very early version of the project selection website, which involved users having to navigate to multiple pages to perform a selection, which makes for a terrible user experience. This was improved upon in order to minimize the number of actions a user must take to perform their desired output.

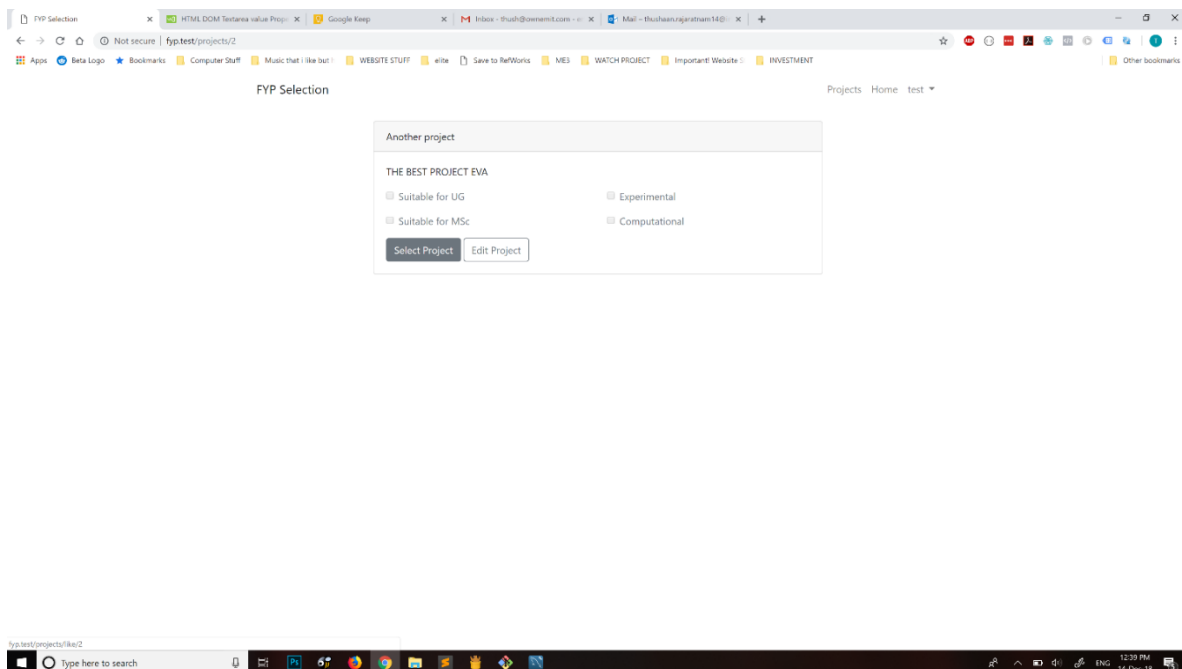


Figure 25: Screenshot of an early version of the individual project view (now replaced by "modals") in January

5.3 VULNERABILITY TESTING

There are numerous ways in which a website can be tested for security flaws, but due to the nature of the author having a bias it makes testing difficult – in the sense that any major security flaws that are deemed as “known” will have an active effort towards being fixed. More interesting is third-party testing for security flaws since usually vulnerabilities that are not known can be detected.

In the case of the FYP website as there was very little resources in the form of finding high level vulnerability testers, the author had to rely on free online resources as well as testing within his group of friends who were familiar with website security.

Using online testing tools such as Pentest yielded positive results about the security strength of the website as shown in Figure 26 (Pentest-Tools, 2019). Any security risk regarding HTTPS was ignored during the testing since this normally occurs due to the website not having a domain name, which was deemed unnecessary previously in the report. Having ignored risks pertaining to the HTTPS, the website was found to be relatively secure, with only a few low-level risks. These risks included the factor that the server technology stack is openly visible which attackers can then use when trying to find potential weakness. Another security weakness was the presence of a robots.txt file, which would only be a problem when used incorrectly to try and hide URLs from users.

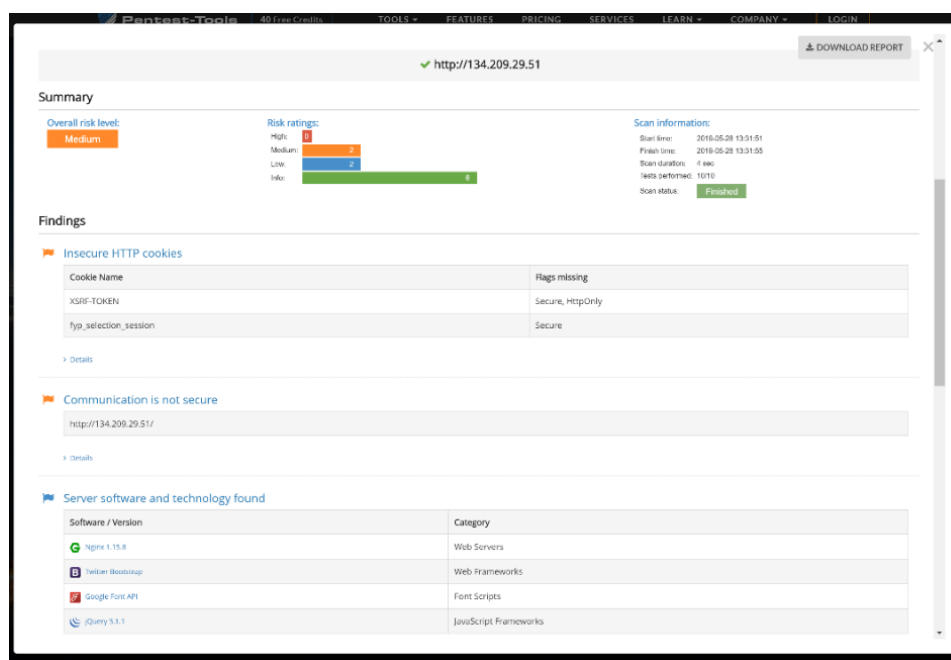


Figure 26: Screenshot of security testing using pentest-tools.com (Pentest-Tools, 2019)

By chance, one of the author's friends also ran some attacks without the author's knowledge, which is the ideal type of test in the sense that there was no bias.

The friend tested SQL injection but rather than only submitting an SQL string into submission form, he also modified the form's headers, an attack which would be usually successful if inputs are not sanitized properly. But having using PDO-binding to prevent SQL injections, the input data was first cleaned by the website, which prevented the attack even though it was somewhat more sophisticated than the typical SQL injection attack. The attempt at hacking the website also involved some mild XSS attacks by putting in some scripts. Due to properly escaping output strings (using Blade as described in the security section previously) these were unable to penetrate the website, as seen in Figure 27.

test

Supervisor: Amis

`<script>alert("boom")</script>`

☒ Suitable for UG ☐ Experimental

☐ Suitable for MSc ☐ Computational

☐ Hidden

Delete Project Edit Project

Number of students having selected this project: 0
Students who have selected this project and their rank:

Figure 27: Project description showing malicious script inserted but unsuccessful in running on website

Another vulnerability detected was an issue with Nginx (the webserver software the website was running), where data is comprised but the solution which would be to run a modified HTTP/2 listener would overwhelm the CPU (Central Processing Unit) usage. The modified listener would be compatible with a more recent version of the Hyper Text Transfer Protocol and immune to such vulnerabilities. This sort of attack is related to a DDoS (Distributed Denial of Service), in the sense that it is an inherent risk that affects many websites (including much larger website with greater resources than the FYP website which will only serve a handful of students) making it very hard to defend against. But given the fact that the website will serve such a small number of students (relative to the millions of users using other such sites), it will mean very serious attackers are unlikely to target the site.

5.4 DEVICE COMPATIBILITY TESTING

In order to ensure the website worked and appeared as intended on a range of different browsers, the author tried to access and use it from browsers such as Google Chrome, Apple Safari, Internet Explorer and Mozilla Firefox which are the most popular browsers used worldwide. The same test was also conducted by accessing the website from computers, laptops and mobile phones of different screen sizes which was also successful, an example of a mobile phone view of the website is shown in Figure 28.

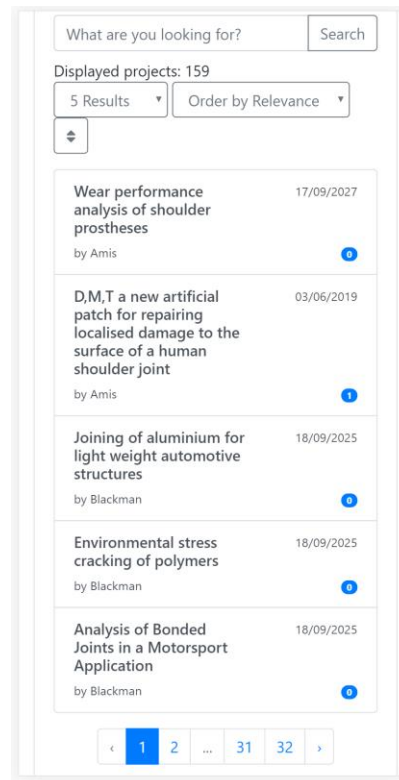


Figure 28: Screenshot of mobile phone view of website

Again, similarly to the vulnerability due to the lack of extensive resources (e.g. not having access to an exhaustive list of devices to test with), the author had to also rely on free online services to test the device compatibility of the website. Using tools such as Browshots.org (Browsershots, 2019) shown in Figure 29, the author was able to see how different devices would render the website.

The test showed that the website was capable of being rendered successfully in a variety of different browsers as well as on different systems (i.e. different operating systems). This will ensure that if not all, a large majority of students should have no issues using the website from their different devices.

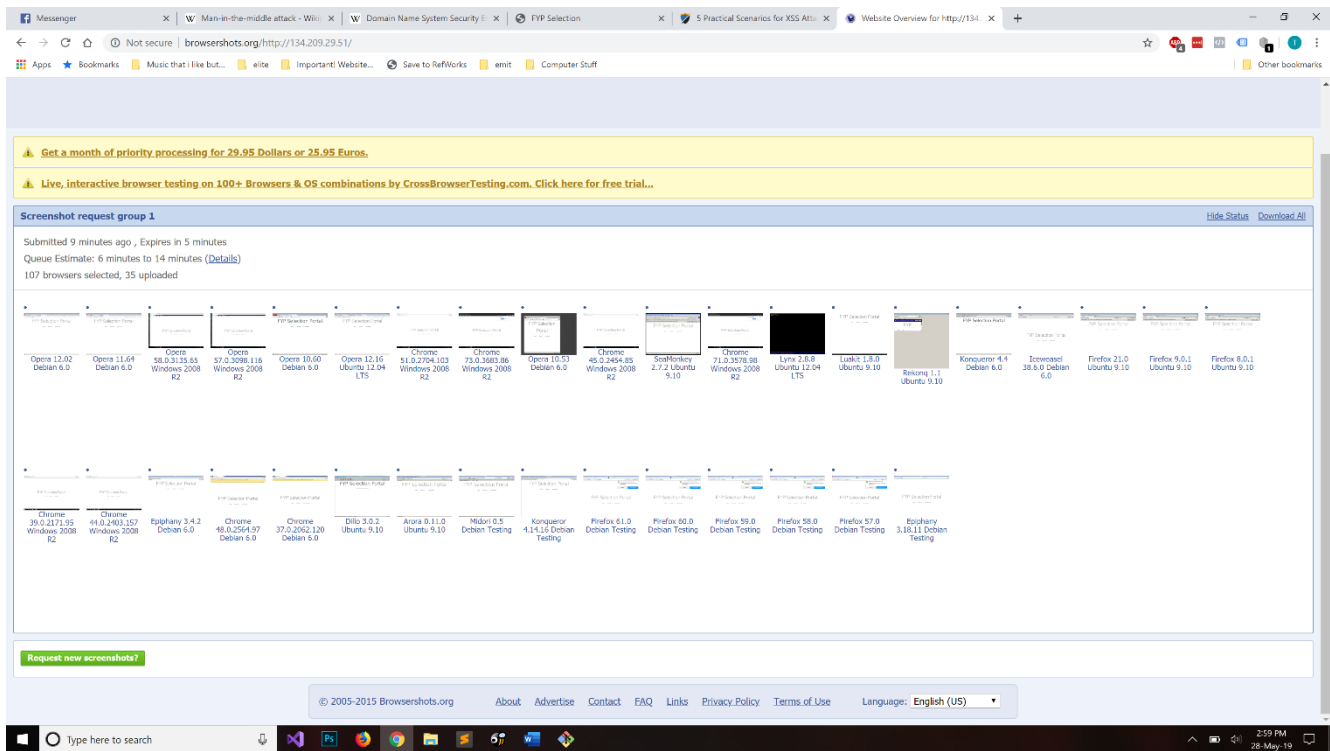


Figure 29: Screenshot of browsershots.org testing for device compatibility (Browsershots, 2019)

Also, thankfully due to using bootstrap's very elegant UI elements (specifically the way the framework handles columns and rows) the website can neatly and fluidly adjust to screen size changes – which is principally what enabled it to be used in a mobile format.

6 DISCUSSION

It should be noted that though the website is functional there is still a lot of work that can be done, which is the nature of software in general as there is always a great deal of improvement that can be done both in terms of efficiency (better student-project matching procedure or website fluidity) as well as UX/UI. Firstly, discussed are the main successes and failures of the project so that the foundation upon which improvements can be made is understood in Section 6.1. Then, information on how the website can be implemented by the university is elaborated on in Section 6.2. Finally, possible further work on improvements that can be made to the website are also elaborated in Section 6.3.

6.1 SUCCESS & FAILURES OF PROJECT

The main success of this project was to be evaluated based on recreating the entire final project selection website. This included building features such as allowing staff to create and manage projects, and students to select and rank projects, so staff could then select students. Finally, the website also had to allow the superadmin (Ambrose Taylor) to be able to export and manage all these selections.

Although the website was recreated, there wasn't much change made to the process of project selection, where the current selection procedure is that all students who selected a project as first choice and received it are first informed, and then students who selected a project as second choice and received it are informed and so forth.

From the literature review, many options of using algorithms to optimize project selection were investigated, but it was found through communication with users (staff and students) and especially through discussion with the project coordinator that the process was better left manual. Since almost 80% of students get allocated their first-choice project, and the remaining 20% of students preferred emailing supervisors to sort their alternative projects, an algorithm which automatically matches them would not be ideal. Therefore, the main goal of the project became to recreate the platform on which this selection process occurred, such that it provided a better experience to staff, students and the project coordinator.

Following this, it was deemed that the project was successful in delivering a fully function final year project selection website which enabled all stakeholders to perform their desired tasks. Testing revealed that current final year students found the website intuitive to use and secure against traditional vulnerabilities, but that there was room for improvement as well.

The main failure of the project was the lack of relevant user testing and feedback. It is very important to highlight the difference between just any feedback and relevant feedback. It was identified early on that methods such as getting surveys from students and staff are not very effective at identifying their actual underlying needs. Since surveys are biased towards a particular set of questions, they don't necessarily allow users to convey a pain-point they would be experiencing when using a service. It was also difficult for users to rate the current system on an abstract scale since there was no other project selection service, they could compare it against.

Nevertheless, some surveys were still carried out, and as expected didn't yield conclusive results. The most useful type of feedback was actual user testing of the website with students and staff. When this was done, rapid feature development and bug detection was possible due to there being many aspects of design and security which the author had not accounted for. Unfortunately, there could have been a lot more done in terms of getting users and staff to use the website, but an inherent problem with this method is that there is very little incentive for non-final year students to simply use the website when it is not needed for them at the moment. This led to the surveys being ineffective.

It is therefore very important that in the future more useful user testing be done – where students and staff are incentivized to use the website to find bugs, errors and improvements based on their firsthand experience of the website. The exact form of the incentivization would need to be discussed further with the University depending on the amount of resources they are willing to put in. A form of getting many students would be to do an Amazon voucher give away (sponsored by the university) for any students who find any errors with the website.

6.2 IMPLEMENTATION BY UNIVERSITY

Although the website has been developed, it is still yet to be integrated with Imperial College's system which would enable students to directly access it as a link sent out on their Blackboard accounts or emails. Integration with the university's systems would be necessary to populate the website's database with the list of staff and students every year, along with their login details. This integration was not done as part of the project since it was determined that the more valuable output would be to develop an entire website without having to spend a large amount of time communicating with Imperial College's IT (Information Technology) team. This can be done as a continuation of the project but will not be technically difficult as it principally relies on ticking a list of administrative requirements. The author will hand over the entire source code (all code related to the website) to the coordinator who can then decide the best way to proceed with integration, either as another Final Year Project or a summer research program for an undergraduate student.

The source code for the website is currently stored on Github (which shall be given by the student to Dr. Ambrose Taylor once the project is over), and due to the very streamlined technology stack involved in hosting the website (as mentioned in earlier sections), the

university should be able to follow this report and easily create an environment where the server and website can be hosted. In short, the university would need to create a Linux server running the above-mentioned requirements in Section 3.1.1 such that the Laravel framework can operate and bring up the website.

The university will then need to spend some time implementing a few components to make the website integrate with their data more easily. The first component mentioned below in Section 6.2.1 is the authentication system, after which all users (staff and students) will be able to sign onto the website using their university login details.

Finally, the university should also spend some investment in getting a relevant domain name (e.g. www.imperialMEngFYPselection.ac.uk) and then securing it with a proper SSL certificate, the importance of which was elaborated in the Sections 3.1.4 and 3.4 of the report.

6.2.1 Authentication System Integration

Integrating the website's authentication system with Imperial College's SAML (Security Assertion Markup Language) system is an important aspect of the project that needs to be done as part of its continuation. The SAML system is a protocol which allows for exchange of a standardized record of user logins, so the user logins into one portal (Imperial College), which then sends the website server data about what this user's email, password and student or staff status is (Imperial College London, 2019). This will make the login process more seamless for users, preventing them from having to register a new account to access the website.

There are packages on GitHub that provide frameworks for easily integrating a Laravel based website's authentication system with another system that adheres to the SAML protocol.

Work on integration of this is heavily important since the university has policies in place that prevent hosting of university student data offsite (i.e. it would go against university policy to have student's account data stored in a separate non-university server).

6.2.2 Project Data Integration

In order to integrate the website's data and functions with the university's servers, the IT team or the administrator who takes ownership of the website can create Algolia search accounts, integrate their project data with the server's SQL databases. The data integration can occur by forming database connections between the website and the university servers, or by using

an existing API (Application Programming Interface) which is a set of communication protocols that allow external applications (such as this website) to access the university's servers and functions.

6.2.3 Administration of Students' Selection

Some additional further work will need to be done with the mechanical undergraduate office as well as the project coordinator to establish the exact fields needed to be exported from the project selection website. This will ensure that the administrative staff can properly use the data for assigning students and projects on their systems. This process can be avoided if the FYP projection selection website directly feeds in to the UG office's database using an API as mentioned above in the data integration section.

6.3 FUTURE WORK

At the end of the project, based on testing and feedback received from the supervisor, a range of improvements were determined which could be implemented to the website in the future as an expansion of the work performed.

6.3.1 Further AJAX work

In order to make the website seem even more fluid to the user, AJAX can be used to make the search, selection and deselection of projects a simultaneous process on the front-end and back-end. This means that the user will not see the page refresh every time they search or select a project, and instead just perceive the whole process as being seamless. There are also generally many other components of the website that can use AJAX for smoother and faster actions, such as student selection by staff and even the loading of project pages themselves. Although, it must be noted that the development of the website would become far more complicated as the frontend of the website becomes heavily dependent on advanced JavaScript techniques (js.foundation, 2019).

6.3.2 Further Search Engine Work

The search engine itself can be improved to have more features since it is the main component of the website that would be used by the students. This includes features such as searches that autofill, learn from user behavior and detects trends to then suggest projects (DigitalOcean, 2019).

6.3.3 Tagging System

It was noticed early that a system of “tagging” or allowing for “categories” of project topics would make it easier for students to find certain project groups. For example, it greatly benefits students if they can search all projects within the “category” of “tribology”. This can be partially done now by using the search function, but would be far more accurate and fluid if the projects were pre-assigned these categories by the staff for when the students search for it (DigitalOcean, 2019). Although, this would require staff to put in more effort by adding tags and categories to the projects for this feature to be useful. Nonetheless, there would be no harm in adding such a feature that provides staff with options to add such descriptions.

6.3.4 Images and Videos

Following on from some of the feedback from students (after they used the website), many mentioned that they would like to see staff upload embedded videos and images to describe their projects. This again would involve a great amount of work to make the website capable of handling such features. The main component that would need to be built is to allow the server to store files (rather than only database entries as it currently does), as well as needing to increase the server storage capacity and access speeds. Fortunately, the university does have a great deal of experience managing such system already, especially with systems like Blackboard which already hosts many files (BlackBoard, 2019).

Consideration will also need to be given to the server having to perform security checks on any uploaded files, as they pose a greater risk of spreading viruses by being stored on the server or spread to a range of students. Although since it would only be staff at the beginning who would be allowed to upload files, it greatly reduces the chance of any malicious actors intentionally uploading dangerous files.

7 CONCLUSION

The project was deemed to be a success since the author achieved the principal objectives of building a website which would enable final year projects at Imperial College London's Mechanical Engineering department to be selected by students, and consequently allocated to them by the relevant staff members. Although the project initially also involved redesigning the project allocation process itself, it was determined that the existing process is highly effective in providing up to 80% of students with their first choice, thus preventing the need for a re-design.

In addition to building an initial website prototype, continuous testing provided an iterative development cycle for the website such that features on design, security and functionality were added based on obtained user and supervisor feedback. The development of the website was done using back-end programming frameworks such as Laravel, and front-end packages such as Bootstrap and jQuery which provided a scaffolding of features on top of which the author could rapidly build parts of the website. The website was first hosted on the author's local computer, as detailed in the report, and then transferred to an online server (Digital Ocean) to be made available to users.

Testing also revealed that there are several components of the website that can be improved, particularly in terms of features such as smarter searching abilities and the uploading of images and videos which would enhance the user experience. The author also realized there was a need for a more effective testing methodology which would incentivize users to provide more vested and therefore reliable feedback. The suggested improvements of testing and potential future work on the project were detailed in the Testing and Future Work sections.

The next step of the project is to integrate the website within Imperial College's systems such that it adheres to all existing college data policies and can exchange information with their databases on user information. This process is also outlined in the report. If desired, another project can continue the work done by the author to improve the website using this report; details on the building blocks of the site, and where all the code is stored are on a GitHub repository, which is to be handed over to the project supervisor.

8 REFERENCES

- Abraham, D. I. R. a. M. D., 2007. Two algorithms for the Student-Project Allocation problem.. *Journal of Discrete Algorithms*, 5(1), pp. 73-90.
- Anwar, A. a. B. A., 2003. Student project allocation using integer programming. *IEEE Transactions on Education*, 46(3), pp. 359-367.
- BlackBoard, 2019. *Blackboard | Education Technology & Services*. [Online]
Available at: <http://www.blackboard.com/>
[Accessed 20 10 2019].
- Browsershots, 2019. *Check Browser Compatibility, Cross Platform Browser Test*. [Online]
Available at: <http://browsershots.org/>
[Accessed 25 4 2019].
- DigitalOcean, 2019. *Droplets Digitalocean*. [Online]
Available at: <https://www.digitalocean.com/docs/droplets/>
[Accessed 20 4 2019].
- Eriksson, B., 2012. https://en.wikipedia.org/wiki/Document_Object_Model#/media/File:DOM-model.svg. [Online]
Available at: https://en.wikipedia.org/wiki/Document_Object_Model#/media/File:DOM-model.svg
[Accessed 6 4 2019].
- Frey, R., 2010. *Diagram of interactions within the MVC pattern..* [Online]
Available at:
<https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller#/media/File:MVC-Process.svg>
[Accessed 20 3 2019].
- Garfinkel, G. S. a. S., 2002. *Web Security, Privacy & Commerce: Security for Users, Administrators and ISPs*. 2nd Edition ed. Sebastopol: O'Reilly Media.

HashiCorp, 2019. *Vagrant Documentation*. [Online]

Available at: <https://www.vagrantup.com/docs/>

[Accessed 5 11 2019].

Imperial College London, 2019. *User accounts and passwords*. [Online]

Available at: <https://www.imperial.ac.uk/admin-services/ict/self-service/connect-communicate/user-accounts-passwords/>

[Accessed 15 3 2019].

js.foundation, 2019. *jQuery API*. [Online]

Available at: <https://api.jquery.com/>

[Accessed 20 11 2018].

Kazakov, D., 2001. *Coordination of Student Project Allocation*, York: Computer Science Department, University of York.

Leggett, F. S. F. a. F., 2018. *Imperial 4th best university in UK; problems with student satisfaction continue*. [Online]

Available at: <http://felixonline.co.uk/articles/2018-05-04-imperial-4th-best-university-in-uk-problems-with-student-satisfaction-continue/>

[Accessed 20 10 2018].

Maatwebsite, 2019. *Laravel Excel Documentation*. [Online]

Available at: <https://docs.laravel-excel.com/3.1/getting-started/>

Manlove, D. a. O. G., 2008. Student-Project Allocation with preferences over Projects.. *Journal of Discrete Algorithms*, 6(4), pp. 553-560.

Mark Otto, J. T. a. B., 2019. *Bootstrap documentation*. [Online]

Available at: <https://getbootstrap.com/docs/4.1/getting-started/introduction/>

[Accessed 27 11 2018].

Microsoft Corporation, 2016. *Documenting your projects on GitHub*. [Online]

Available at: <https://guides.github.com/>

[Accessed 20 1 2018].

Nicolas Dessaigne, J. L., 2019. *Algolia Documentation*. [Online]

Available at: <https://www.algolia.com/doc/>

- Nixon, R., 2014. *Learning PHP, MySQL & JavaScript: With JQuery, CSS & HTML5*. 4th ed. s.l.:O'Reilly.
- Oracle Corporation, 2019. *MySQL 8.0 Reference Manual*. [Online]
Available at: <https://dev.mysql.com/doc/refman/8.0/en/>
[Accessed 10 12 2018].
- Oracle Corporation, 2019. *MySQL Workbench Documentation*. [Online]
Available at: <https://dev.mysql.com/doc/workbench/en/>
[Accessed 10 12 2019].
- Oracle Corporation, 2019. *Oracle® VM VirtualBox® User Manual*. [Online]
Available at: <https://www.virtualbox.org/manual/UserManual.html>
[Accessed 20 11 2018].
- Otwell, T., 2019. *Laravel*. [Online]
Available at: <https://laravel.com/docs>
- Pentest-Tools, 2019. *Website Vulnerability Scanner - Online Scan for Web Vulnerabilities | Pentest-Tools.com*. [Online]
Available at: <https://pentest-tools.com/website-vulnerability-scanning/website-scanner>
[Accessed 25 4 2019].
- Pinto, D. S. a. M., 2011. *The Web Application Hacker's Handbook: Discovering and Exploiting Security Flaws*. 2nd ed. Hoboken: Wile.
- Rajaratnam, T., 2018. *ME4 Project Plan Report: ME4 Project Allocation Process and Website/Software*, London: Mechanical Engineering Department, Imperial College London.
- Rajaratnam, T., 2019. *ME4 Progress Report: ME4 Project Allocation Process and Website/Software*, London: Mechanical Engineering Department, Imperial College London.
- Ristić, I., 2017. *Bulletproof SSL and TLS*. 2nd ed. London: Feisty Duck.
- Romero-Medina, A., 1998. Implementation of stable solutions in a restricted matching market.. *Review of Economic Design*, 3(2), pp. 137-147.

Shema, M., 2012. *Hacking Web Apps*. 1st ed. Amsterdam: Elsevier.

Taylor, A., 2018. *ME4 INDIVIDUAL PROJECT GUIDELINES FOR STUDENTS*. [Online]

Available at: https://bb.imperial.ac.uk/bbcswebdav/pid-1362674-dt-content-rid-4757093_1/courses/DSS-ME4_MPRJ-18_19/ME4%20Project%20Guidelines%202018-2019%282%29.pdf

[Accessed 15 October 2018].

Wodehouse, C., 2018. *A Beginner's Guide to Back-End Development*. [Online]

Available at: <https://www.upwork.com/hiring/development/a-beginners-guide-to-back-end-development/>

[Accessed 30 11 2018].