University of Plymouth

School of Engineering,

Computing, and Mathematics

PRCO304

Final Stage Computing Project

2019/2020

University Sports League Management System

James Shobbrook

10585576

BSc (Hons) Computer and Information Security

Acknowledgements

I would like to sincerely thank my two project supervisors assigned to me in development of this project.

Liz Stuart, for the early stage guidance and motivation to produce sections of my project in a timely yet thoughtful manner. The early stand-up meetings set me on a clear path for the rest of the project, and I am thankful for the encouragement she shown our group.

Shirley Atkinson, who took over from Liz Stuart as project supervisor, for the excellent job she performed in guiding our group through a difficult time in the world, and always being available for any queries or questions I had. The effort made by Shirley, allowed me to be able to produce my project to the best of my ability and I will always be grateful for this.

Abstract

This report describes the process that was carried out in the development of a university sports league management system, built with the intention of improving online interactivity from sports club members.

The report begins with an analysis into current league management systems, to paint a picture as to what key features and shortcomings exist in the market. The results of this analysis, paired with an investigation into suitable development technologies, are used to form client objectives. The technologies chosen to develop the application were the Laravel framework paired with a MySQL database implementing Test Driven Development. Legal, social, ethical and professional issues are presented with a notable focus on the security measures implemented into the application.

Requirements are presented followed by the project design that outlines how the application is intended to function. The development process is then described, showing the testing processes undertaken and the progress that was undertaken throughout each sprint interval. Following this is the end-project report that evaluates the produced project against both the client objectives and the project development objectives, showing that the project met the targets set out earlier in the report.

A project post-mortem is carried out that reflects on the technologies chosen and steps taken, evaluating whether the decisions taken were beneficial to the project or not. The project overall was a success, leading to increased knowledge into a new field and meeting all client objectives set.

Table of Contents

Word count: 10811

GitHub Repository Link

1. Introduction

1.1. Background

Sports within any university is an integral foundation, used to allow students to interact with each other and form relationships. These relationships not only improve student's sporting experience but also their overall university experience and mental well-being. 33% of university students often or always experience feelings of loneliness (Pereira, 2020), as technology progresses and face to face meetings reduce, ensuring these vital relationships are still formed and maintained can help to reduce this figure. This can be accomplished through increasing the online tools available for students to be involved with their sports club.

The ability that a student currently to interact with their sports society online, is primarily through social media platforms such as Facebook and Instagram. These platforms serve the purpose of posting news and communication within the club. External applications are used to view results and upcoming fixtures for the club called league management systems. These systems are often used only by captains of a club and information is passed onto the club through social media platforms.

Creating a platform that combines the news and communications of a university sports club with a well-functioning, enjoyable to use league management system, would create an environment for club members of inclusivity that would encourage more interaction whilst preserving the professional atmosphere that a sports club portrays.

1.2. Problem Identification

The current measures in place for students to monitor and interact with their team's progress have various issues, these can hinder their sport's experience and therefore overall university experience, these issues are:

- Outdated, often limited to mobile or desktop exclusively, difficult to use software to monitor fixture results. Causing students to rely on a select few members of a club for information.
- Data often updated manually using spreadsheets. The lack of automation means large amounts of time are used when updating results and fixtures. The ability to make mistakes is also increased when data is entered manually.
- Lack of involvement with software other than to check their latest results and fixtures. There is a need for extra features to improve user involvement and add to the university sports experience.

1.3. Client Objectives

Presumed objectives of potential clients have been produced as there is not a live client for this project. The objectives for the league management system are to:

- ❖ Increase accessibility to features of a university sports team on the internet for all students.
- ❖ Create features that allow for increased interactivity between students, to create a more personalized and welcoming experience than existing systems.
- ❖ Streamline the process an administrator has in creating and updating information on fixtures and results.

1.4. Project Aims and Objectives

Considering the problems identified and the client objectives. The primary aim of the project is to develop an interactive sports league management system targeted at universities, that involve minimal administrative manual input to manage.

In order to meet this aim, the following objectives have been set:

a) Investigate current sports league management systems and analyse the features that are used in creating the user experience.
b) Create a set of requirements that use the results of the research to fulfil the client objectives set out.
c) Research and analyse the development technologies that would be best suited to meeting the requirements.
d) Develop the application using the chosen technologies whilst adhering to an attentive yet flexible project management methodology.

2. Analysis and Investigation

Being sure that this project was going to contain all the functions that a league management system required was important. This was because there was not a live client that could set out requirements for the project to hit and the features that were needed. Analysis of some current league management systems was undertaken, chosen due to popularity and relevance to a university setting, with the aim of being able to form the basis of requirements for the project.

2.1. Analysing the product domain

Without a client, the significance of ensuring that analysis of what the current league management domain looked like and how they functioned was key to ensuring expectations were kept high whilst ensuring that the scope of the project does not grow too large. Failing at this stage in the process would more than likely of resulted in either setting requirements that were either too small and simple or too large and unrealistic.

2.1.1. Existing League Management Systems

League management systems were evaluated alongside each other in order to analyse some key functionalities that were shared, which provided a core of what was features were essential to a league management system. Three sports league management systems were examined and analysed.

- BUCS Play Mobile App
- Leisure Leagues
- Goals

The three applications were analysed individually through visiting the application or website and using as much functionality as was available. Unfortunately, it was not possible to get full access to all the websites features as they require the user to create and pay for a team. For one website this problem was solved by being given access to a full account that was a member of a team to allow full access to view all the features of the system. The features of the systems were reported and analysed against each other to see the key shared and unique functionalities of each system.

All three of the applications create an experience for the user that focuses on the league standings, fixtures and results as the focus, showing the importance of making sure the information that users initially register to the site for is available. Some of the shared functionalities of all the applications that form the core that is needed for the project to follow:

- A league table for every division that records data from all results in that division.
- A list of upcoming fixtures specific to that division as well as recent results.
- Ability to register as part of a team through making an account on the website.

It was noted that in all the applications, it was difficult to find only the information pertaining to the user's team and only that team.

2.1.2. Interviews

Two main interviews took place to help form some of the requirements for the project. One interviewee was a Plymouth University student who was a player of a university sports club, the other a team captain.

Points gathered from the initial research were put forward to the interviewees as well as asking them what some flaws with current league management systems were and some features, they think would be beneficial for a user of a league management system. The key points to assist in forming requirements came that from the interview:

- Give players a custom profile that has information about themselves on it.
- Team page with ability to make posts that the public can see

- Mobile friendly and Desktop friendly responsiveness of key pages a player or captain would need.

2.2. Investigation

Once a baseline of requirements for the project were formed it meant that methods of how those requirements would need to be decided on. It was important to ensure that research was investigations were made into the methods of development, so that the chosen methods were best suited to the needs of the application.

2.2.1. Development Methodologies

The software development methodology for this project was Agile. This was clearly the best software development methodology to use for the following reasons. Agile allows projects that have a short lifespan to completed using sprints, these allow the development to be undertaken incrementally to focus on outputting a minimal viable product. This is important to ensuring that the essential requirements are set out, helping to mitigate the risk set out in my project vision of the scope of the project becoming too large and running out of time. Other benefits of the Agile methodology that were beneficial to add to the project are things such as:

- Keeping a product backlog of features to add
- Test Driven Development
- Focus on interactions over processes

2.2.2. Evaluation of Possible Technologies

Various types of application for the project were considered such as a mobile app or desktop application. A web application was chosen as the system that would be used to develop the project due to its accessibility at any point on any device. This is important for sports players as they will not always want to use a computer to keep updated on matches but also will want the immersive full screen experience that viewing an application on a computer can give you. Something that needed to be strongly considered when making this decision, is the application would have to be developed using mobile first software development ideals so that it would function for both players and captains when using it. For administrative duties it is assumed in the project that a desktop will be used rather than a mobile phone.

After deciding that a web application would be used. Research was undertaken to identify which languages and technologies would be best suited to the project. It was important to consider whilst investigating the technologies that the development team is one person so if skills needed to be trained in a technology, that the documentation and resources for it were well thought out, accessible and easy to learn. Due to this it was narrowed down quickly to either a PHP Framework or

ASP.Net as the main technology for developing the project. Although the author has prior development experience in C#. After reviewing the options Laravel PHP framework was chosen. This is due to its popularity notoriety as the most popular PHP framework in the world (Dudkin, 2020). It is extremely well documented and has accessible tools to learn the framework.

Because of the interacting classes of the application it was clear that the choice in databases needed to be a relational one, to allow for the functionality of the application. Due to previous experience MySQL was chosen as the database for the project.

Other technologies that will need to be utilised are HTML, CSS, JavaScript for front end web development as well as jQuery/Ajax to allow for interfaces on the application to change dynamically as a result of user interaction

2.3. Developing Skill in Technologies chosen

Various learning tools were used to develop skills in the technologies that were chosen. Three weeks prior to starting development was allocated to hone skills needed. Resources used to learn technologies were:

- **W3Schools**: A free website used for basic skills development and assistance for HTML, CSS, JavaScript and jQuery/Ajax
- **Laracasts**: A website the has clear concise 'from scratch' tutorials on Laravel that assisted in learning all the basics of the framework.
- **Stack Overflow**: Used in skills development to discover solutions to problems that were difficult to fix in the skills development stage.
- **YouTube**: tutorials and new functions of MySQL, Laravel, jQuery were learned through YouTube videos.

3. Method of Approach

3.1. Agile Scrum Development Framework

The development framework chosen for Agile software development was the Agile Scrum framework. Scrum is not a methodology, it replaces a programmed algorithmic approach with a heuristic one, with respect for people and self-organization to deal with unpredictability and solving complex problems (Schwaber, 2020). This statement identifies the key reasons why it was chosen. It allows flexibility and changes rather than a rigid set road that needs to be followed exactly. This choice as the project progressed based on flexibility as a primary reason, was clearly shown to be a good one as the Coronavirus pandemic worsened and progress on certain steps of the project were interrupted. Using scrum, different sections of the project with access to could be worked on e.g. documentation, so

time was not wasted. If a different framework such as a Waterfall approach would have been chosen it would have been difficult to make the changes to what was being worked on based on current situational factors.

Agile scrum could not be fully implemented in this project. This is because usually when using the framework there are roles assigned to different members of the team. These being product owner, scrum master and the development team. However due to this being a solo project this was not possible. As a result, certain other features of scrum couldn't be implemented such as daily progress meetings to check progress on sprints. The framework had to be adapted to suit the needs of the project on an individual level. The following attributes of the Scrum Methodology were applied in the project development:

- **Sprints**: making progress on the project at each interval that can be seen or reviewed by anybody. E.g. Adding a new feature, Setting up environments.
- **Product Backlog**: Keeping a list of tasks based on priority to be developed and released.
- **Sprint Backlog**: Creating a list of product backlog items to be completed within the sprint period.
- **Meetings**: Stand up meetings were held weekly where progress was reported with project supervisor and peers initially. These meetings became fortnightly as development progressed and moved to online Zoom meetings when it became impossible to hold stand-up meetings due to the university closing because of the Coronavirus epidemic.
- **Test Driven Development (TDD**): Working tests were written for each feature and the code was then adapted so that it would pass the tests that ensure that the code is functioning correctly.

3.2. Laravel

Laravel was chosen as the primary language for this project. It is a framework that implements the Model, View, Controller (MVC) architecture which allows for a separation of resources in development, therefore making the application more efficient.

One of the main reasons choosing Laravel is that it attempts to take the pain out of development by easing common tasks used in most web projects. This is especially true and was discovered whilst developing certain aspects of a web application that are usually time consuming and tedious. Through the inbuilt authentication scaffolding that Laravel offers. Encrypted, safe, functional authentication for the web application was able to be quickly implemented to allow for more time to be spent developing bespoke features for the application. As well as authentication, other features of Laravel were used to assist in the security driven development of the application, such as:

- **Policies**: Classes that are used to structure authorisation around specific resources or models in the application.
- **Middleware**: a mechanism for filtering requests sent to an application and redirecting based on authentication such as a user group.
- **CSRF Protection**: Simple to implement CSRF tokens for forms on the webpages to protect from cross-site request forgery attacks
- **Blade**: Templating engine allowing for organised layout pages, simple syntax as well as auto escaping all variables to increase security.
- **Eloquent ORM**: Object-relational mapper that allows for simple connection and relationship definition between objects in application and database.
- **Model Factories**: A tool that seeds customisable random test objects that persist in the database. Saves time manually creating test data whenever needed or when database is wiped
- **Artisan Console**: Command-line interface used with Laravel to perform actions such as creating models, views, controllers. Features Tinker which 'allows you to interact with your entire Laravel application on the command line' (Otwell, 2020).

3.3. Test Driven Development (TDD)

Test Driven Development is achieved by creating a working requirement of the project in the form of a test. The code for the application can then be refactored so that it passes the test that was created. In doing this you can be sure that the application functions correctly.

The Project was tested through the command-line. When multiple tests were written for the application this would mean that all tests are ran every time. This creates an overriding positive for TDD because development can check every single requirement of the application individually after every change made. The result of this is quickly being alerted to what part of the application is no longer working, fixes can then be made to what has been changed. If TDD was not implemented in this situation, the developer would not be aware that the change that they have just made has broken functionality in another section of the program. When the developer eventually finds the functionality is not working, they will not be instantaneously aware of what has caused it.

Some other benefits of using TDD that were experienced during development of this project were: (Winter, 2020)

- Increased productivity through improving focus, due to focusing on writing code in order to pass a test.
- Deeper understanding of requirements. Knowing what needed to be achieved to consider the requirement met.
- Increased confidence in making changes. This was important as it allowed development to not be held back by worry that another feature would stop

functioning correctly and the entire application could stop functioning with no reason as to why.

4. Legal, social, ethical and professional issues (600 words)

When personal information of users is collected and used in an application there are many legal obligations that must be adhered to. As well as legal issues there are also ethical, social and professional issues that can arise as a result of any project. A sports league management system does present some, although not many obstacles to tackle as a developer in comparison to a banking application or children's application.

4.1. Legal Issues

'The Data Protection Act 2018 controls how your personal information is used by organisations, businesses or the government' (UK Government, 2019). This act is the UK's implementation of the General Data Protection Regulation (GDPR) which is the EU law that states the various principles that must be followed when it concerns user's data.

Some of these principles relevant to the Sports League Management system are that the information: (UK Government, 2019)

- used for specified, explicit purposes
- handled in a way that ensures appropriate security, including protection against unlawful or unauthorised processing, access, loss, destruction or damage
- used for specified, explicit purposes

It was important to consider these principles when designing and developing the application. The application stored the following personal information to comply with the law:

- Email Address
- Username
- Full Name (If applying to a team)

This information is used for a specific purpose by the application to manage authentication, list the player name as a part of a team, handle sending confirmation emails and password reset emails.

All other information about a user is chosen to be created and added to their profile page e.g. Player Bio, Position.

4.1.1. Security

As mentioned above, a principle of data protection that needs to be abided is that information is 'handled in a way that ensures appropriate security, including

protection against unlawful or unauthorised processing, access, loss, destruction or damage.’

Although in the application there is not a large amount of personal information stored. Information security was considered when developing all features of the application. Some of the following security measures were implemented to improve information security:

4.1.1.1. Authentication

Information was made more secure by putting access to information behind a layer of authentication. All accounts were accessed by a personalised login with a password stored in the database that is hashed with Bcrypt password hashing.

Only when logged into own personalised account can that account be managed. The only exception to this is that an administrator of the site can manage non personal information regarding to an account such as team status or deleting posts made.

Whenever actions are being performed that involve the authenticated user, to ensure that actions made by user is the currently authenticated user, the Laravel Auth facade is used throughout the application. The facade ‘provides a "static" interface to classes that are available in the application's service container’ (Otwell, 2015). This allows the application to find the authenticated user when creating new content as well as check to see if the user performing the action is the currently authenticated user.

4.1.1.2. Authorisation

Ensuring that the application has strong levels of access control throughout and that users performing actions are authorised to do so is of paramount importance when it comes to Information security of the application.

Without correct authorisation, users would be able to perform actions that they should not be able to. Resulting in the possible loss, leak or destruction of personal information. Enough steps were taken in the development of the application to ensure that data protection principles are not mishandled. One step taken to assist authorisation was using policy classes.

Policies ‘are classes that organize authorization logic around a particular model or resource’ (Orwell, 2020). For each model in development a policy was written. This allowed for straightforward access to change or test authorisation for a specific requirement. Within the policy are methods for the actions that are being attempted to be performed on a model (e.g. Adding a new team). Written into these methods within the policy are the checks that need to be completed before the action is started (e.g. Checking that the user belongs to the admin group). If the policy is not met completely then the action is cancelled and the user is shown a 403 Unauthorized error.

Whilst authorisation overall is important for information security. One drawback of using policies in the application development was that it could be difficult to test different parts of the application, without needing to alter code in the controller to remove the policy request.

The use of middleware was another layer of authorisation that was used in the application. ‘Middleware provide a convenient mechanism for filtering HTTP requests entering your application’ (Orwell, 2020). Like policies middleware was used to authorise user requests. However, unlike policies, instead of showing the user a 403 error. Requests made that triggered an action from middleware would redirect the user to a different page. If the user was unauthenticated and attempted an action that needed an authenticated user, the request would redirect to the login page.

Three other core middleware were developed for the application. Player, Captain, Admin. These middleware were implemented and attached to routes that required authorisation for actions. If a Player or captain attempted an action meant only for an admin such as visiting the admin panel. Instead of being shown a 403 error, first they would be redirected back to the previous page, improving user experience of the application. Only if the middleware failed on some actions would the policy trigger.

Having the two layers of authorisation of policies and middleware, created a level of security that is robust meaning that if one method of authorisation failed. The other method would most likely capture the request.

4.1.1.3. Cross-site Request Forgery Protection (CSRF)

Keeping information safe means protecting against various attacks on the application to gain unauthorized access to features of the application.

A CSRF attack ‘is a web security vulnerability that allows an attacker to induce users to perform actions that they do not intend to perform’ (What is CSRF, 2020). For example inducing a user to visit a malicious website whereby clicking a button on the site sends a post request to the legitimate website from the victim, using their authenticated credentials to create a new administrator account for the attacker to gain access to the application. Giving the attacker the ability to steal information resulting in a data protection violation.

The developed application contains protection from this attack using a CSRF token. If a post request is made to the application there is a requirement of a CSRF token that is generated for the user attempting to perform the action in the correct manner. The token is then compared to the required token and if there is a mismatch of token the user is redirected to a 419 error page.

4.1.1.4. SQL Injection and XSS protection

In order to prevent SQL Injection, the application is developed with SQL requests being made through Laravel Eloquent ORM. When requests are made to the database using Eloquent requests parameter binding is used to ensure that SQL requests cannot be made to the application that perform malicious actions such as a

‘WHERE 1=1’ injection, in which all rows from a table can be returned from a request (SQL Injection, 2020). When parameter binding is used these requests will always be escaped meaning they will not function.

XSS protection is implemented in the application through the blade templating frontend Laravel provides. ‘By default, Blade ‘{{ }}’ statements are automatically sent through PHP's ‘htmlspecialchars’ function to prevent XSS attacks’ (Otwell, 2020). So that no malicious script cannot be inputted and ran on the application.

4.2. Social and Ethical Issues

4.2.1. Misuse of Application Features by Users

Due to the ability of team members to make posts, profiles and other personalized content on the webpage, the application could be used in an unethical way that is inappropriate and could offend other users of the web application such as posting inappropriate pictures, spam or even illegal content. In order to mitigate this risk. Site administrators can remove any content posted to the application from any member.

In order to post the user must have signed up to a team. This creates a line of accountability for the misuse by linking the offender to a team which has a captain in the event of repeat offences.

4.2.2. Underage Users

As the application is intended for university students there is little consideration for children using the application. However, the application is developed with the awareness that people under eighteen can attend university in the UK and therefore an age restriction of over 18 is implemented to avoid the ethical issues that arise with children using the application

4.2.3. Competition Between Teams

Although healthy competition between teams is permitted within the application it is important to make sure the application does not become a platform for a toxic environment between teams competing.

4.3. Professional Issues

The universities that sign up to the application must give permission to the application to use its logo, results and player information. This is clearly the largest professional issue that needs to be addressed. Teams must give permission manually so that the administrators of the site can populate the information.

5. Project Management

As an Agile Scrum project management method was used for development of the project. The steps that needed to be taken specific to the project, that will allow value

to be demonstrated at the conclusion of each stage of the project, were able to be to be planned and undertaken. Therefore, adhering to the key Agile principles.

The following Agile project management features were implemented in the completion of this project:

- **Project Vision:** Explaining from a high level perspective. What the aims and objectives of the project are and the problems that the project's purpose is to address and potentially solve.
- **Risk Plan:** Addressing the key risks that could befall upon the project at an early stage so that measures can be put into place to mitigate these risks before they cause severe complications that affect the deadline of the project. Although key risks were considered it is important to note that the risk of a global event causing disruption to ability to run the project as planned was not listed in the risk plan. This was an oversight that came to fruition as the project was being developed and disruption to workflow was caused. Extensions were applied to deadlines which meant that the loss of time due to disruption due to global events was noticeable but not overwhelming.
- **Weekly Sprint Reviews:** Ensuring at the conclusion of each sprint a review was conducted to address what of value was created in that sprint. This allows future sprints to be conducted in a way that maximise efficiency of producing value.

5.1. Trello

Trello was the tool that was used for project management for the duration of development. Trello uses a card system that is simple and easy to edit. The cards contain tasks to be completed as part of the development.

In the early stages of development, the tasks to be completed were posted in a low level manor that was more technical. This as development progressed resulted in requirements being forgotten and needing to be addressed later. After sprint 7 (Appendix 2) the decision was made to convert all tasks into user stories format. This allowed development to be conducted on tasks based specifically on requirements for the project and decreased the need to readdress tasks that were missed out. This decision was made although in a late sprint, at a time when development of the application was at its most productive. Therefore, time lost due to mis-organisation of tasks was minimal.

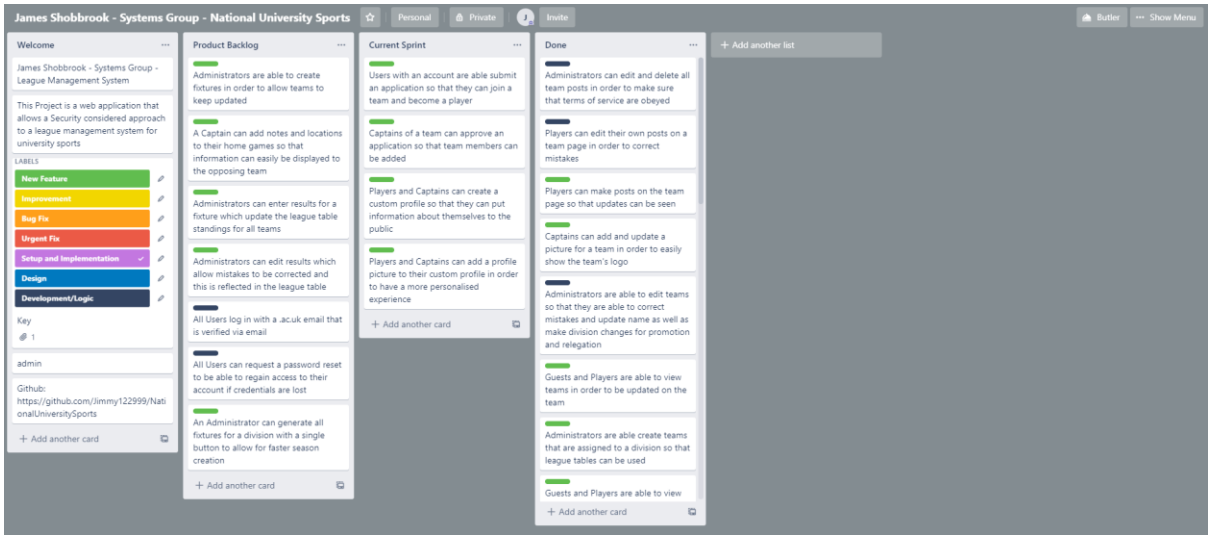


Figure 1: Project Trello Board

5.2. Version Control

The version control used for the project was GitHub. GitHub is a platform that is used host code, review changes and manage a project. As the development team used in this project was a solo developer, it was unnecessary to use the GitHub branch system for multiple developers to make changes. A master branch was used to commit all changes.

The ability to backtrack any changes to a previous version and clone the most recent version was vital to not majorly disrupting the project. In March 2020 an error with uploading to an FTP server led to all local files for the project being overwritten and lost. The GitHub version was able to be cloned and the project faced no hinderance.

6. Requirements

As there was no live client for this project. Requirements had to be developed based on what each user-group on the website would want to be able to use the application for. Ensuring that the requirements were formed around ensuring a minimum viable product (MVP) would at least be completed.

6.1. Requirements

The requirements were developed with the user-groups on the application as the centre of attention. The requirements were split into Essential and Desirable requirements. The MVP would be achieved if all the essential requirements were hit. If only the essentials were hit, although producing a viable product, would not satisfy the developer as showcasing extra functionality was an aim throughout development. The Essential/Desirable system was however very useful for prioritisation of development time, allowing the key principle of Agile project management of producing a feature of value at every interval to be completed. Full list of requirements can be found in Appendix 3

6.2. Non-Functional Requirements

As well as the requirements needed that display what the intended application does. Non-functional requirements address some of the requirements needed that allow the application to meet all the functional requirements.

Some non-functional requirements addressed in the project are:

- **Reliability:** Ensuring tasks are performed how they are intended. Using test driven development this non-functional requirement was addressed, as all functionality can be tested with a single command
- **Performance:** Ensuring functions are completed quickly and efficiently. Achieved by making sure code is well optimised, especially database queries, and there are not large data redundancies.
- **Usability:** Ensuring functions and requirements can be fulfilled by a user in a simple and clear way. This is addressed with the minimalistic design with administrative actions being performed alongside the standard application layout in most cases, such as creating a new sport or division.
- **Compatibility:** Ensuring features of the application function on various platforms such as mobile/desktop difference and different types of browsers e.g. Firefox does not accept the 'Date/Time' HTML input so an alternate needed to be decided for picking a date and time.
- **Consistency:** Ensuring the application has a layout that is consistent across the entirety. If not kept consistent the application will become confusing and can affect the usability.

7. Design

After requirements had been finalized and prioritized. The design stage of the application began. Software design is "is a process to transform user requirements into some suitable form, which helps the programmer in software coding and implementation" (Tutorials Point, 2020). The stages of design were carried out in order to make clear:

- What components were needed to meet the requirements.
- What actions those components needed to perform.
- The relationships between the components.
- A base for what the user interface should look like.

7.1. UML Class Diagram

A class diagram “is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects” (What is a Class Diagram, 2020). It was identified as a key step to add to the design stage as it addresses three of the four aims of the design stage.

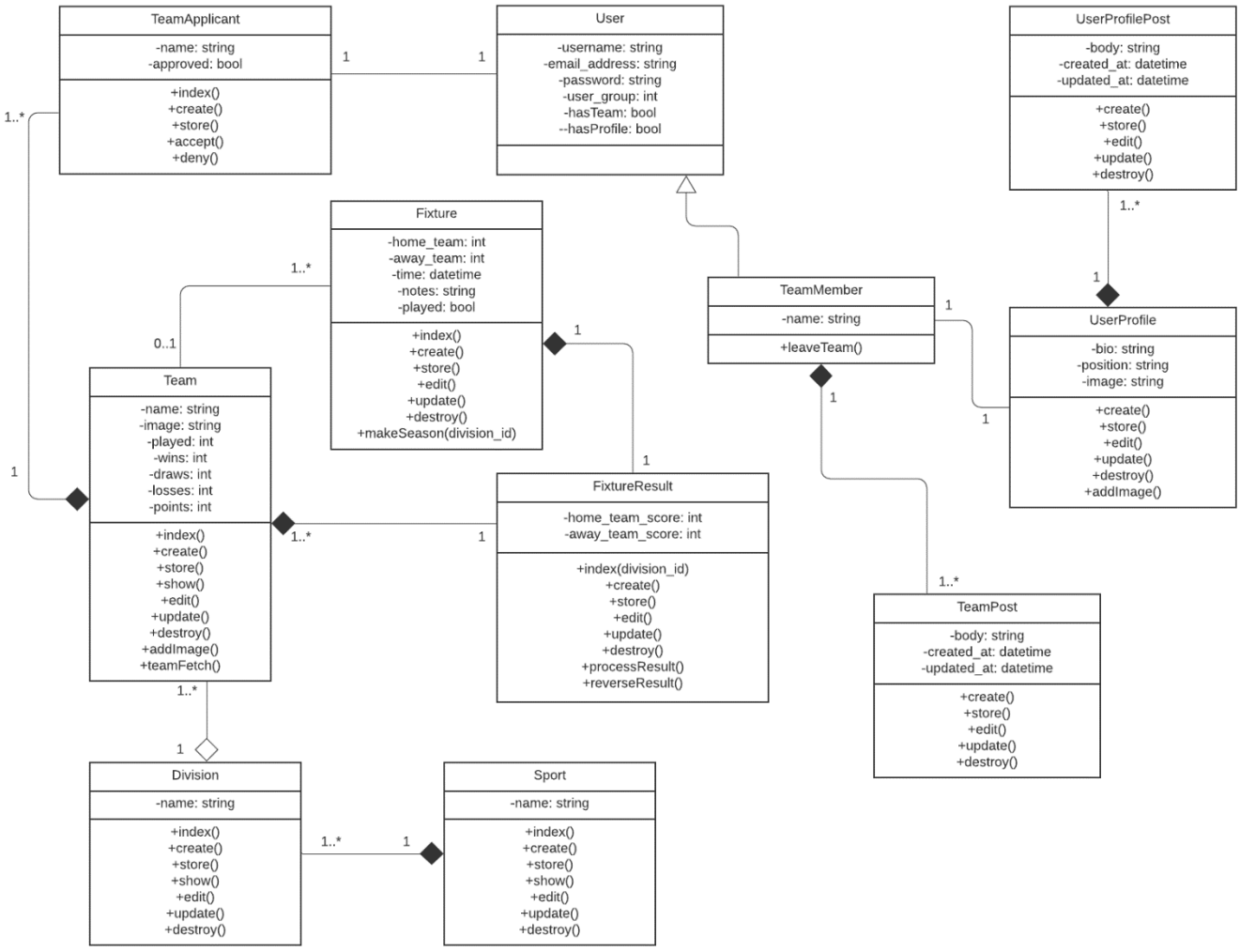


Figure 2: Unified Modeling Language class diagram of proposed application

7.2. Database Design

After the classes were finalised an Entity Relationship Diagram was produced that defines the relationships between classes.

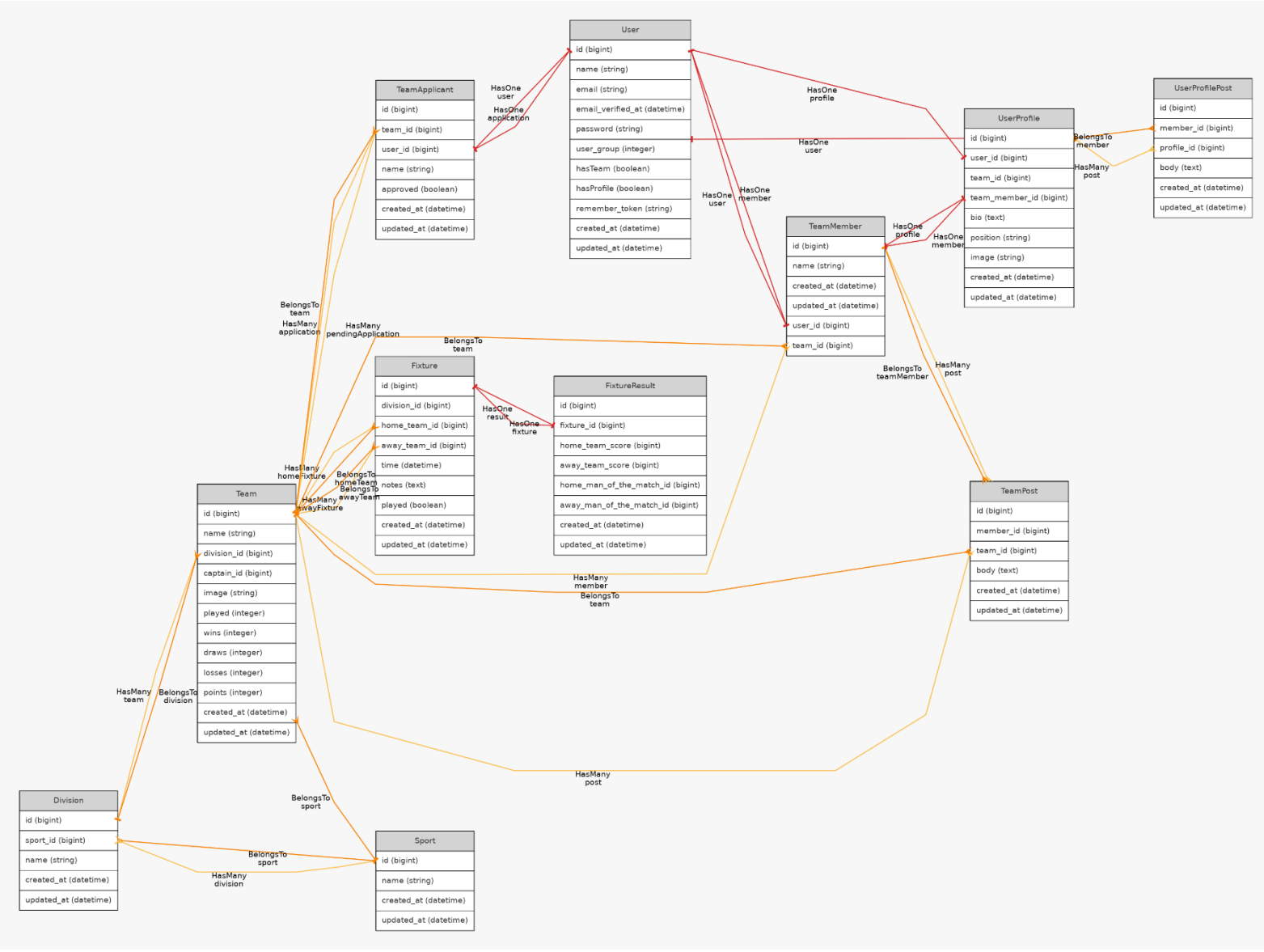


Figure 3: Application Entity Relationship Diagram

7.3. GUI Designs

Prototype wireframe models of what the features of the application were and how they would be implemented from an aesthetic point of view were developed (Appendix 4). The reason for developing these were to see any problems that may have arisen from implementing the requirements without needing to spend large amounts of time developing features that have clear issues.

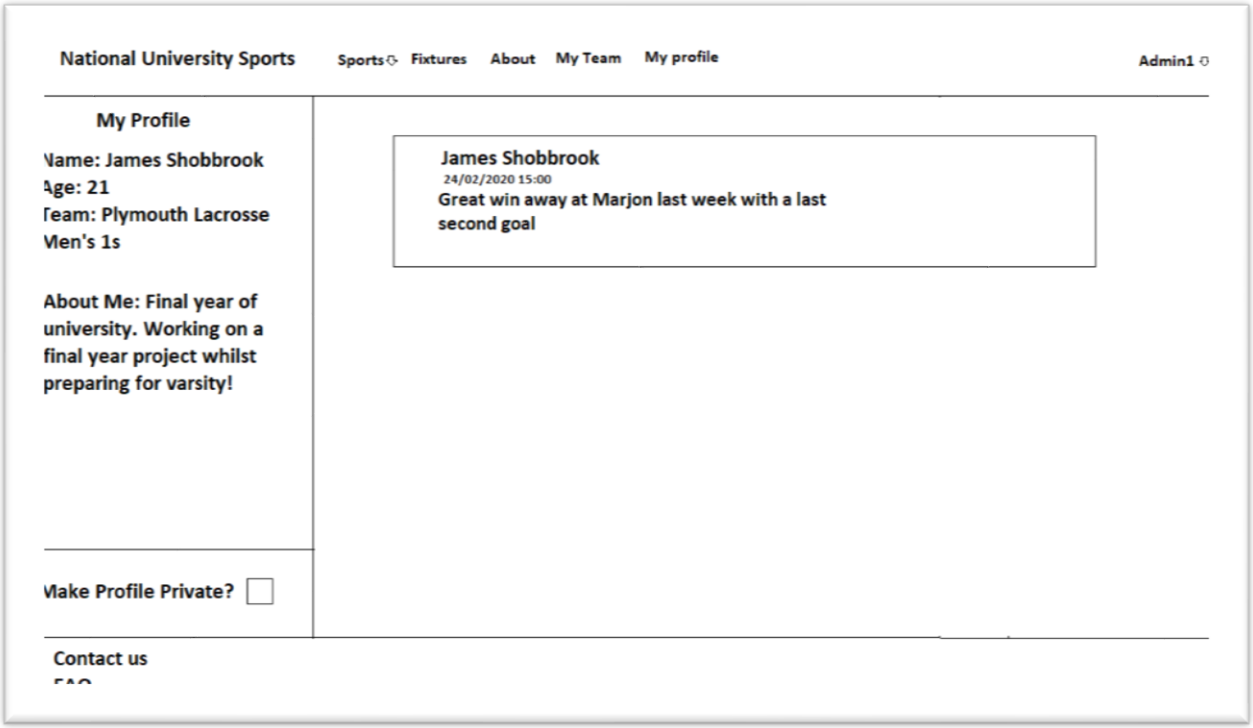


Figure 4: Wireframe design of the user profile page

These wireframes were created to stand as a foundation for which the developer could reference when creating the pages. They were not the intended final application, as it was clear that features would be added on top of the wireframe design at a later stage e.g. Profile pictures.

8. Development and Testing

After the research, investigation, analysis and design stages had made progress. The development of the project was able to begin. The development stages were split into 9 sprints over 4 months. The sprints varied in length; the first 5 sprints were a week in length each. The final 4 sprints were spread over 2 or 3 week periods. These differences in timeframes were useful as in the early stages of development as there were a greater number of tasks that were not as time consuming as later in development. This meant the progress that was reported weekly had enough content produced after very sprint.

8.1. TDD

Agile software development recommends using Test driven development noting “many teams report significant reductions in defect rates, at the cost of a moderate increase in initial development effort” (Agile Alliance, 2016). This was very apparent throughout different stages in the development but overall which fixing breaks in the code. Using TDD saved a large portion of time in development overall.

Test driven development was implemented in this project using PhpUnit which is included in a default Laravel installation. When first attempting to implement testing there was an issue that caused PhpUnit to not run correctly and no tests to be

conducted. This was later found to be an issue with the incorrect version of PHP being installed on the system. When PHP 7.3 was installed the default placeholder tests ran correctly and testing could begin to be implemented to the development of the actual application.

The next challenge that was faced was when database functionality was needed to allow the tests to be correctly populate and assert the tests. With the initial setup when running a test that would create a record and assert that the record was created. Actual data was being created in the database of the application. Although when running one test this was not a large issue. When scaled up with the main benefit of unit testing, that is testing all previous functionality. It caused too many records to be created and the database to become clustered.

```
<server name="APP_ENV" value="testing"/>
<server name="APP_KEY" value="base64:HTDzpje7l7YTu4IWJqih/gp+6CQnleEHcpeEE3ZY6iE="/>
<server name="BCRYPT_ROUNDS" value="4"/>
<server name="CACHE_DRIVER" value="array"/>
<server name="DB_CONNECTION" value="sqlite"/>
<server name="DB_DATABASE" value=":memory:"/>
<server name="MAIL_DRIVER" value="array"/>
<server name="QUEUE_CONNECTION" value="sync"/>
<server name="SESSION_DRIVER" value="array"/>
```

Figure 5: Testing Environment for PHPUnit Testing

One possible fix to this would be to delete the record after every test. However, this approach would cause lots of traffic to the main database and was viewed to not be the most effective approach. A separate testing environment was set up that would run the tests. SQLite with the database stored in memory was the approach chosen. This was because SQLite allowed SQL queries to be ran which is important for testing functionality of the application, but it does not need a live connection to a database. The ability to use memory as the database meant that the speed in which tests could be conducted compared to a MySQL live database was vastly quicker, allowing for tests to be ran for the whole application after even a small change meant that the rigidity of the application, throughout development was solid.

In order to run any test that required database entries throughout the application, the two foundations that needed to be set out were firstly, the initial data that was needed to perform all the tests for the model. This was achieved with a function that would run SQL queries that would create sample data that is the same as the data that would be found in a live run of the function (Figure *). Secondly in order to test thoroughly it was important to make sure that authentication was tested. The actingAs method allows a “way to authenticate a given user as the current user” (Otwell, 2016). This method was used along with a model factory (See section 3.2) to ensure that when conducting a test. It is not only testing the functionality of a method but also the authentication and authorisation permissions of every feature.

```php
private function populateData()
{

    Sport::create([
        'name' => 'test'
    ]);

    Division::create([
        'name' => 'test',
        'sport_id' => 1
    ]);

    Team::create([
        'name' => 'test',
        'division_id' => 1,
        'played' => '0',
        'wins' => '0',
        'draws' => '0',
        'losses' => '0',
        'points' => '0',
    ]);

    TeamMember::create([
        'name' => 'testMember',
        'team_id' => 1,
        'user_id' => 1,


    ]);

}
```

```php
private function isAdmin()
{

    $this->actingAs(factory(User::class)->create([
        'user_group' => 1
    ]));
}

private function isCaptain()
{

    $this->actingAs(factory(User::class)->create([
        'user_group' => 2
    ]));
}


private function isPlayer()
{
    $this->actingAs(factory(User::class)->create());
}
```

Figure 6 (left) and 7 (above): Populating the test database for the User Profile tests (6) and the 3 different test authentication methods (7).

When creating the tests for a model in the application such as the ‘Team’ model. The most important tests that needed to be implemented for every model were the four CRUD functions (Create, read, update and delete). Tests of extra functionality were added where required but ensuring the CRUD foundation was covered, allowed for development of tests to be in a structured orderly way. When naming a test, a tag needed to be placed above the function to allow PhpUnit to read the test. Names of test methods were written in a manner that, when tests were performed. An error message pointing to the failed method would give clear information as to what being tested had failed, without the need to manually find the method and check, which may have been needed if methods were not written in a long form way.

```php
/** @test */
public function A_Member_Of_A_Team_Can_Create_A_Profile()
{

    $this->isPlayer();
    $this->populateData();


    $response = $this->post('/profile/create', [
        'user_id' => 1,
        'team_id' => 1,
        'team_member_id' => 1,
        'position' => "Goalkeeper",
        'bio' => "test"
    ]);
    $this->assertCount(1 , UserProfile::all());

}
```

Figure 8: Unit test creating a user profile and asserting it now exists

When a specific test for functionality is written. Firstly, the foundations are set by creating the authenticated user to run the test. Followed by populating the database to run the test. It is important the user is created before the data is populated as some of the created data needs a user to reference. Secondly the function being tested is written and if applicable saved to a variable. In figure * it demonstrates a post request being sent to a URL to create a user profile. In the live application this would be the form data that is sent when submitted. As there is a test database the test will, if successful, create the profile in the temporary database. After the test has been fully ran an assertion is then made to the database to see if the test was successful of not. In figure * the number of profiles in the temporary database is asserted as 1. As there were no profiles before the test was ran. If there is now 1 profile, it means the profile was created and therefore the test was successful.

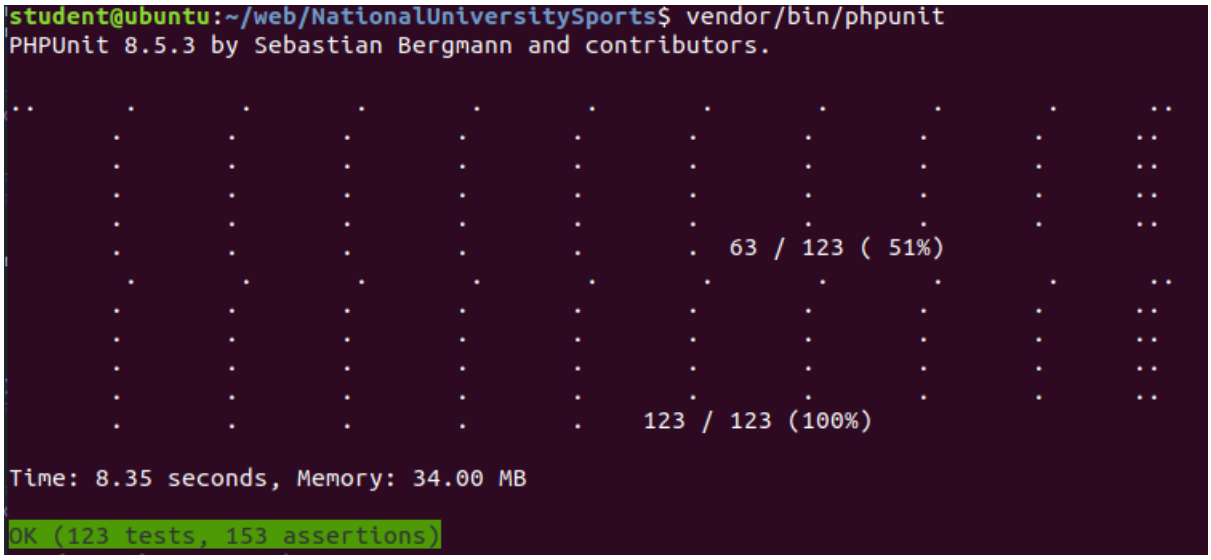


Figure 9: Console showing how tests were ran during development

Overall 123 tests were written for the application that made a total of 153 assertions. This number of tests allowed for a level of assurance when developing future features, that any indirectly caused issues to other parts of the application, would be shown in a clear concise way that pointed to the problem. The trade-off of initial time spent developing tests for time saved when developing more features was considered throughout development.

8.2. Verification and Validation

Ensuring that as the project was being developed, verification and validation of what was required was an important aspect considered. As there was no live client for this project, the requirements developed (Appendix 3) were used as the base to verify and validate what was being developed with what was needed.

Verification is “testing that your product meets the specifications / requirements you have written” (Martin, 2015). Verification measured were implemented through reviewing the code and reviewing what had been developed at the end of every

sprint, checking against the requirements for the application to ensure what is being built is what is required.

Validation is the test of “how well you addressed the business needs that caused you to write the requirements” (Martin, 2015). This was addressed in the project through functional testing of the application when it was completed to ensure it performed exactly as it could Unit tests throughout development were used for validation. UML diagrams were also referenced to ensure that what was being developed functioned as intended.

8.3. Functional Testing

Functional testing is “is a type of software testing that validates the software system against the functional requirements/specifications” (Guru99, 2020).

Unit testing as mentioned above was one method of functional testing that was implemented throughout development. Other forms of functional testing were also employed to ensure rigidity of the application.

Component testing is the process of “testing a module or component independently to verify its expected output is called component testing” (Shah, 2020). This testing method was used for ensuring that the UI of the application functioned correctly and for testing the performance of pages loading time. From a security perspective, SQL injection tests were also performed on various models of the application to test that correct development practices were followed.

8.4. Usability Testing

Usability testing is the process of “evaluating a product or service by testing it with representative users. Typically, during a test, participants will try to complete typical tasks while observers watch, listen and takes notes” (Assistant Affairs, 2020).

As the application is developed for university sports teams. Three different University of Plymouth students were recruited for usability testing:

- A regular club member of a University of Plymouth sports society.
- A first team captain of a University of Plymouth sports team.
- A president of a University of Plymouth sports society.

Having three different test users with different knowledge and ideas on how a university sports league management application should function was important for the monitoring of the usability testing.

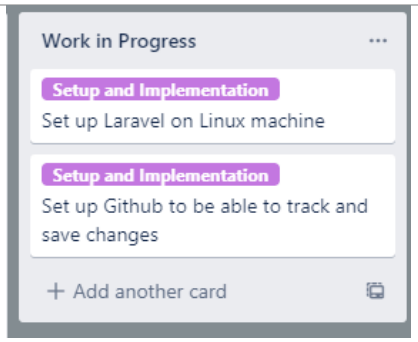
The initial plan for performing usability testing on the application was to set up the application on a portable environment and meet the three test users together and monitor each testing the application whilst answering questions and making notes. This plan had to be refactored after global events led to face to face meetings not being possible. The main problem faced with performing the usability testing was how to give the testers access to the application and monitor them. Having the testers set up an environment on their own computer seemed impractical. The

approach taken was to purchase a web domain and host the application online for the testers to access. The testing took place on a Zoom video call in which each tester would share their screen of themselves performing actions that were read out by the developer. The main changes that came of the usability testing were UI related, ensuring that a main theme was followed throughout the application.

8.5. Sprint Reviews

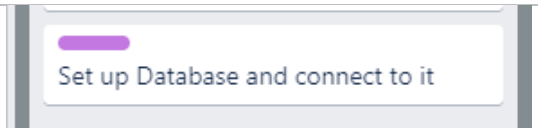
At the conclusion of each sprint a review was conducted that contained a summary of actions and a list of features developed.

8.5.1. Sprint 1

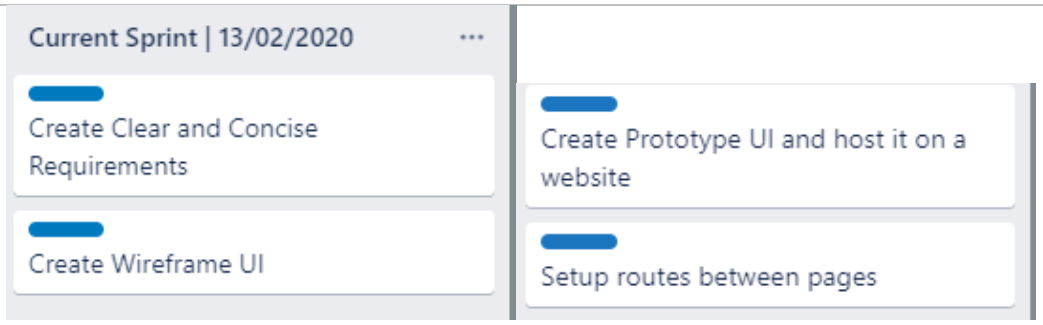
Duration	24th– 31st January
Sprint Backlog	
Summary of actions	The focus of sprint 1 was to set the foundations that were needed for future development of the project. The Trello board was created, and task cards created. ensure the virtual machine was able to produce a working Laravel installation and that the project would be able to make commits to a GitHub repository through the local environment. The stand-up meeting this week focused on setting a rough outline of what the project was about and the problems it was trying to address.
Features Developed	❖ Trello Board Created ❖ Product Backlog Populated ❖ Working Laravel local environment ❖ Connected Project to GitHub

8.5.2. Sprint 2

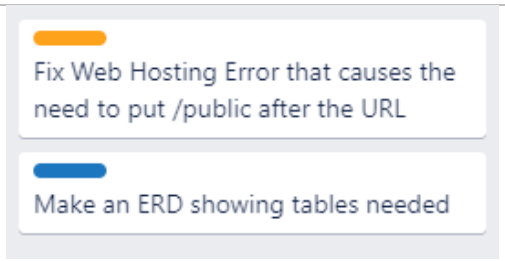
Duration	31st January – 7th February

Sprint Backlog	
Summary of actions	In previous projects databases and connectivity to them had caused the developer issues that were time consuming. The sole focus of sprint 2 was to ensure that this task was completed, and database related design and development could proceed. The stand-up meeting was focused on design of requirements and a follow up informal meeting with a project peer was held to discuss what requirements were suitable for each project.
Features Developed	❖ MySQL Installed and Setup ❖ Database Created ❖ Laravel environment linked to database

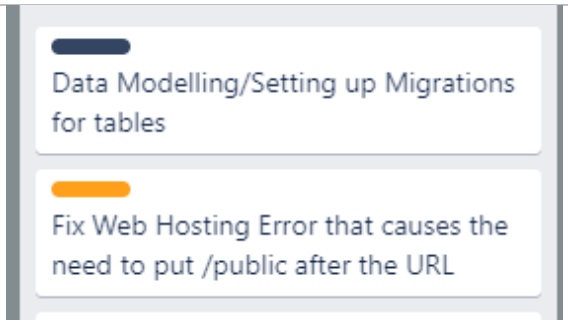
8.5.3. Sprint 3

Duration	7th – 13th February
Sprint Backlog	
Summary of actions	Sprint 3 at the request of the project supervisor focused on finalising requirements for the project. A wireframe UI was developed (Appendix 4). A prototype application was created that began to implement routing between pages. The stand-up meeting set the goal of completing a UI wireframe for peer review at the following meeting.
Features Developed	❖ Requirements Finalised ❖ UI Wireframe ❖ Prototype Application UI with routing implemented

8.5.4. Sprint 4

Duration	14th -21st February
Sprint Backlog	
Summary of actions	Sprint 4 initially had the sole focus of creating the ERD for the application. This is because it was discussed and decided at the previous weeks stand-up meeting that the aim for the following week should be to target an epic which is an agile methodology term for “a big chunk of work that has one common objective” (Yodiz Team, 2016). This week also focused on putting the prototype for the application on a website to access remotely. This created a hosting bug that needed to be fixed. After the ERD was created views pertaining to Sports and Divisions were created with static data and routes between the pages were set up. The stand-up meeting to review the created wireframes was cancelled on the morning of the meeting. Instead wireframes were peer reviewed by email to a project peer.
Features Developed	❖ ERD Diagram ❖ Application hosted on a website ❖ Sports view created ❖ Division View created

8.5.5. Sprint 5

Duration	21st-28th February
Sprint Backlog	
Summary of actions	Sprint 5 focused on creating models for the application that functioned correctly with the database through a migration. A migration is a file that can “modify and share the application's database schema” (Otwell, 2020). Models and migrations for two of the main data containing models were created.

<table>
<tr><td></td><td>Authentication was added to the application that allowed the user to login and logout. A user group attribute was added to the Users table that functioned as shown below:

<table><tr><th>User Group</th><th>Role</th></tr><tr><td>1</td><td>Administrator</td></tr><tr><td>2</td><td>Captain</td></tr><tr><td>3</td><td>Player/Guest</td></tr></table>
The stand-up meeting for this week was cancelled by supervisor.</td></tr>
<tr><td>Features Developed</td><td>❖ Created Sport model
❖ Created Division model
❖ Created Sport Migration
❖ Created Division Migration

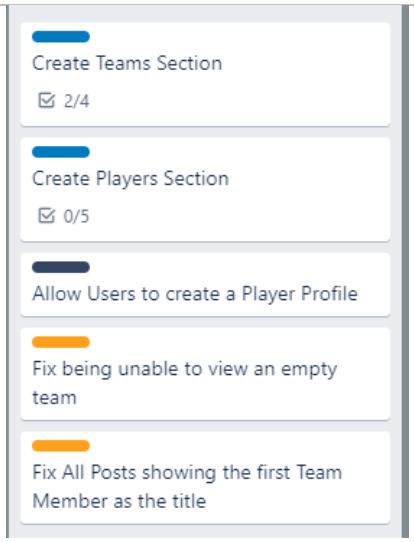
❖ A Guest can create a user account
❖ A User can login
❖ A User can logout</td></tr>
</table>

8.5.6. Sprint 6

Duration	28th February – 13th March
Sprint Backlog	
Summary of actions	Sprint 6 was the first sprint spaced over a 2 week period as opposed to the previous 1 week sprints. It focused on creating controllers for Sports and Divisions that focused on CRUD functionality. This was the first point in which relationships between models were put into use. As the team model is not yet created it isn't possible to view the teams belonging to a division.

	However, the create, update and destroy functionality was completed during this sprint Middleware was put into place to redirect users to three different view folders based on user group. The three different view folders contain different layouts which include buttons only available to an administrator. This will be revisited later to find a solution that allows one view folder for all user groups. The main reason it is needed is that currently every method that is called needs to check for a user group, redirecting to the correct page after performing the function. There was no stand-up meeting scheduled for this sprint.
Features Developed	❖ An Administrator can create a sport ❖ An Administrator can view all sports ❖ An Administrator can edit a sport ❖ An Administrator can delete a sport ❖ An Administrator can create a division that belongs to a sport ❖ An Administrator can edit a division ❖ An Administrator can delete a division ❖ A Player and Captain can view a sport ❖ A Player and Captain can view a division. ❖ Middleware redirects users based on their user group attribute

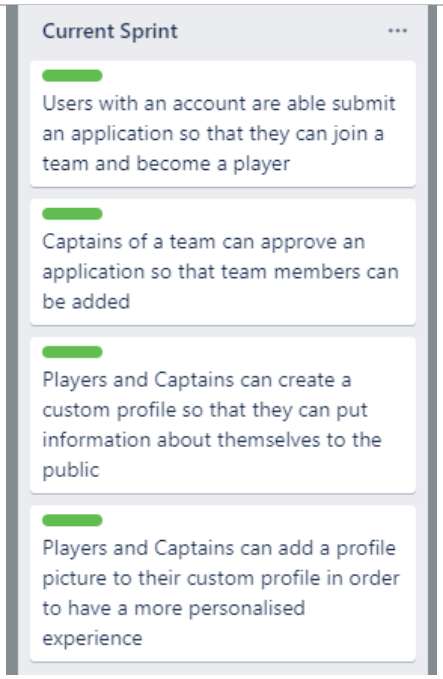
8.5.7. Sprint 7

Duration	13th – 27th March
Sprint Backlog	

Summary of actions	Sprint 7 was another large 2 week sprint which allowed a lot of time for development. As the sprint progressed, development time was interrupted due to needing to relocate where development was taking place as a result of the COVID-19 pandemic. The final few days of development of this sprint saw an increase in productivity due to increased capability of development tools and methods. The focus of sprint 7 was to produce a functional foundation of the application which was as followed: ❖ A team that can belong to a division that can belong to a sport. ❖ A team can contain team members. ❖ A team page that can have posts created by members. ❖ A user profile page in which the user can create posts. Although team members can belong to a team, the ability to join a team is not yet implemented and it currently assigned manually for testing. The ability to make a post as a member on the team page created an issue of ownership of the post always being given to the first member assigned to that team. This was fixed by assigning the post owner ID through the authenticated user's ID rather than team member ID. One error occurring was that pages were unable to load under certain circumstances such as a team not containing members was unable to load. This error was caused due to the controller passing data to the view that contained data that involves team members. This error was resolved using checking that variables are set and if they are not, not performing actions that are impossible. The stand-up meeting was not held due to COVID-19 in person, however it was scheduled as a Zoom meeting. The developer was unable to attend due to it being scheduled whilst relocating.
Features Developed	❖ An Administrator can create a team. ❖ An Administrator can view a team page. ❖ An Administrator can edit a team. ❖ An Administrator can delete a team. ❖ A team member can create a post on the team page. ❖ A team member can edit their own post on the team's page. ❖ A team member can delete their own post on the team's page. ❖ An Administrator can edit and delete all team posts.

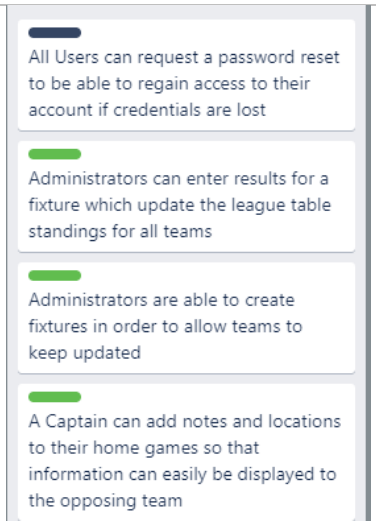
	❖ A team member can create a user profile. ❖ A team member can edit a profile. ❖ A profile owner can create a post.

8.5.8. Sprint 8

Duration	27th March – 24th April
Sprint Backlog	
Summary of actions	After sprint 7 the decision was made to refactor the product backlog into user stories. This allowed for requirements to be the aim of any task completed. Sprint 8's focus was on refining the application so that it functioned without manual inputs. The ability for a user to request a team and for the captain of that team to approve and deny requests was implemented. The ability to add and update pictures for both a team page and profile was added to personalise the experience. This sprint included an overhaul of the three separate view folders with middleware redirecting based on user group. Policies were implemented that allows both authorisation of methods and the ability to display or hide content based on user group.

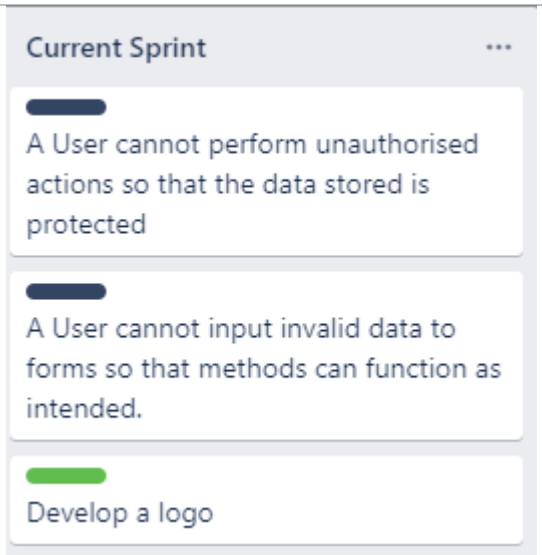
	All views are now stored in a single location and authorisation has been implemented. A stand-up meeting was held on a Zoom call in which progress was updated.
Features Developed	❖ A User can create an application for a team. ❖ An Administrator can view, accept and deny applications for a team. ❖ A Captain can view, accept and deny applications for their own team. ❖ An Administrator can create, update and delete pictures for a team or user profile. ❖ A Captain can create, update and delete their team's picture. ❖ A team member can create, update and delete their profile picture. ❖ Policies created for Sport, Division, Team, Team Post and User Profile models.

8.5.9. Sprint 9

Duration	24th April – 8th May
Sprint Backlog	
Summary of actions	Sprint 9 focused on progressing development to a stage that the application makes it at least to the point of being a minimum viable product and meeting at least the essential requirements that were set. Fixtures were added that allowed results of games against other teams to be updated. This meant that the functionality of a league management system was now working with divisions being able to show the standings between teams in the league table.

	A season generator was created that populates a division with fixtures between the teams in a way that fixtures are evenly spread. This was a process that took multiple attempts and failed multiple times before being completed. Pagination was added to various locations of the application to make the site and the data more user friendly.
Features Developed	❖ An Administrator can create, edit and delete a fixture. ❖ An Administrator can enter and edit a result for a fixture. ❖ A Captain can update notes section of a home fixture. ❖ Division league tables are updated correctly from fixture results ❖ Team page displays team's fixtures ❖ Division displays fixtures and results ❖ Pagination added to team page, profile page, fixtures and results.

8.5.10. Sprint 10

Duration	8th – 15th May
Sprint Backlog	
Summary of actions	Sprint 10 focused on validation and security of the application. This final sprint was used to add form validation for every input on the application. Policies were fully implemented so that authorisation was in effect throughout the application when performing actions. A logo was designed and added to the application for a more custom look.

Features Developed	❖ Policies fully implemented. ❖ Validation added to forms. ❖ Logo added to navigation bar

8.6. Development Overview

Overall there were very few noticeable issues with development that caused loss of time. A large amount of time was spent on design and setup in the earlier sprints. This may have been too much and could have instead spent some of this time on development. Although the strong foundation set allowed for development to progress without much issue. The use of Test Driven Development throughout was a key factor in development as any error could be pinpointed, whereas on previous projects, large amounts of time were spent finding the root of an error.

The project functions as intended and has met all the requirements that were initially set out. This may point to a lack of ambition when initially setting out the requirements although large amounts of time were spent developing to ensure these features were implemented.

9. End-Project Report

The primary aim of the project was “to develop an interactive sports league management system targeted at universities that involve minimal administrative manual input to manage.” Overall the project was a success as the project has met all the objectives set within the deadline set. The only large issue within development was the relocation due to the COVID-19 pandemic, causing loss of time.

9.1. Client Objectives Review

Increase accessibility to features of a university sports team on the internet for all students.

This objective was met using the knowledge of the investigation into development technologies. Choosing a web application gave the system, so long as developing in a mobile first manner, the ability to be accessed and used on both desktop and mobile. When developing one problem that interfered with meeting this objective was cross browser compatibility. The website caniuse.com was referenced whilst developing to ensure that features added would be accessible to a high enough percentage of global browser users to be considered viable.

Create features that allow for increased interactivity between students, to create a more personalized and welcoming experience than existing systems.

The ability for team members to have an area specifically for their team to interact was something missing from the existing league management systems that was unearthed, thorough analysis into the existing systems and the interviews conducted prior development.

This was implemented through a team page that any member signed up to the team can access. The ability for members to post news and information to the rest of the team and the public is something that transforms a mundane league management system into a personalized welcoming network of sports teams and their members.

Other features implemented to meet this objective were:

- A user profile page that members could include information and post personal updates about themselves.
- Ability to upload both team and personal pictures and avatars to represent themselves in a more vibrant way then using only text.

Streamline the process an administrator has in creating and updating information on fixtures and results.

At all stages of development, ensuring that any action that needed to be made to update information, could be made by an administrator of the site. Whereas in traditional league management systems the information was sometimes updated completely manually. This application allows fixtures and results to function as an object that the administrator has complete access to create, edit and destroy using a method that utilises the already created Sports, Divisions and Teams. This reduces the amount of repetition and mass of data that is usually required as all information pertaining to an object is contained and accessible.

The fixture season generator implemented was done so with the primary intention of time efficiency, as the administrator has all the tools with the application to create a season manually.

9.2. Project Objectives Review

Investigate current sports league management systems and analyse the features that are used in creating the user experience.

The investigation stage was performed with the intention of gaining the knowledge of what current league management system's aims are and how they meet these aims with application features. This investigation into the existing systems gave a clear picture of what features would be essential to an interactive system. The interviews were useful in applying the generic league management system knowledge gained to a university setting.

Create a set of requirements that use the results of the research to fulfil the client objectives set out.

Using the knowledge from the investigation and interviews conducted. Requirements were produced that met all the client objectives. As development progressed it proved time consuming to complete the requirements, but they were completed. Some features were added that went above and beyond the requirements specified such as the season generator for a division.

Research and analyse the development technologies that would be best suited to meeting the requirements.

This objective was a success as the technologies chosen allowed the project to progress in a time efficient manner. Laravel provided a platform with which to develop and implement complex features to meet all the requirements created.

Develop the application using the chosen technologies whilst adhering to an attentive yet flexible project management methodology.

The implementation of the application fulfilled this objective using Agile software development practices. Test Driven Development was a tool that allowed issues with the system to be easily fixed and saved development time to work on other features of the application.

10.Project Post-Mortem

Technologies

The technologies chosen as development progressed, were proven to be the right decision. Laravel allowed for large features of the application, such as authorisation and authentication to be implemented with their class libraries. Ability to learn the framework intuitively with Laracasts proved a key factor in development efficiency over time.

MySQL was simple to set up on a local development environment and caused few issues throughout the development. It functioned well when test driven development was implemented at an early stage.

Trello was a tool that was not used in an efficient way at the beginning of the project. It was only during Sprint 7 that it was used in a way that gave value to what was being developed. GitHub was a useful version control method that allowed changes to be reviewed and monitored throughout development.

Agile

Agile proven to be a useful tool for both software development practices and project management. The flexibility that agile has in cases of disruption was useful when global events caused disruption. The stand-up meetings proved useful both in the early planning stages and after meetings were moved onto Zoom.

Design, Development and Testing

The Class and ERD diagrams designed to illustrate how the system would function proved valuable as the number of classes and relationships would have been difficult to keep track of in the absence of a relational diagram. Some functions of the application may have been made clearer to develop if a state diagram had been produced to show the behaviour of the system. However overall design painted a clear enough picture of how the application would function, to develop it with minimal confusion.

Development of the application was made smooth through the Agile approach of producing something of value during each sprint. This method also helped keep motivated during development as progress could be seen in a clear way. The use of user stories allowed requirements of the project to always be at the centre of any development decisions made. Legal, social, ethical and professional issues were considered throughout development as a key requirement. The security measures applied using policies, middleware and authentication. As well as development practices undertaken allowed the application to be at a stage in which it could be released publicly and not be liable for any legal incompetent development practices.

Testing played a key role in the development of the application. Implementing Test Driven Development turned out to be a key decision that saved time and made development less confusing. The ability to see what effect any change would have on the rest of the application allowed decisions to be made in a structured informed manner. Black box testing (Appendix 5) was also carried out at the end of development to ensure that even though unit tests all passed, that errors not covered by the tests were not present when performing actions based on the user requirements and for covering errors that may have been present through the views rather than the controller or routes.

Performance and Reflection

Overall, I am pleased with how I conducted myself throughout this project. I managed to achieve all the objectives I set for myself and more. The fixture generator was a feature of the application that took time and knowledge to perfect. My project management was mostly good as time was managed efficiently, although how I presented the requirements on my Trello board for the sprints up until sprint 7 was unclear. Once refactored for the largest sections of development, project management had no issues whatsoever. I was surprised with the amount of technical skills I acquired from research to deployment and review. Especially with the competence in Laravel I now possess from zero knowledge of the framework before the project began.

11.Conclusion

This project aimed at producing an application that could bridge the gap between league management systems and social media club interaction. The intention of this in a broad sense to target lonely students at university and allow them to interact and

form relationships with sports members in the absence of face to face meetings. Global events have shown that real world contact with others is not always possible so the importance of creating online options is even more important now.

In conclusion the project was a success. Although there was not a live client, the objectives set out were met through the development of an application that allowed a university sports team to keep track of results in their division, whilst creating a personalised interactive experience that helps relationships to be formed through an online setting.

In future the application could add additional features that would capture more aspects of social media, to allow it to progress and remove the need for any sport's social media activity on an alternative application.

12.References

Agile Alliance, 2016. What Is Test Driven Development (TDD)?. [online] agilealliance.org. Available at: <https://www.agilealliance.org/glossary/tdd/> [Accessed 22 May 2020].

Assistant Affairs, 2020. Usability Testing | Usability.Gov. [online] Usability.gov. Available at: <https://www.usability.gov/how-to-and-tools/methods/usability-testing.html> [Accessed 23 May 2020].

BUCS Play Mobile App. Available at: <https://play.google.com/store/apps/details?id=com.playwaze.bucscore> [Accessed 23 May 2020].

Dudkin, I., 2020. PHP Frameworks: Best Framework Comparison List. [online] Skywell Software. Available at: <https://skywell.software/blog/php-frameworks-comparison-list/> [Accessed 22 May 2020].

Goals. Available at: <www.goalsfootball.co.uk> [Accessed 23 May 2020].

Guru99, 2020. What Is Functional Testing? Types & Examples (Complete Tutorial). [online] Guru99.com. Available at: <https://www.guru99.com/functional-testing.html> [Accessed 23 May 2020].

Leisure Leagues. Available at: <www.leisureleagues.net> [Accessed 23 May 2020].

Martin, J., 2015. Verification And Validation: The Difference. [online] SmartBear.com. Available at: <https://smartbear.com/blog/test-and-monitor/verification-and-validation-the-difference/> [Accessed 23 May 2020].

Orwell, T., 2020. Authorization - Laravel. [online] Laravel.com. Available at: <https://laravel.com/docs/7.x/authorization> [Accessed 22 May 2020].

Otwell, T., 2016. Testing - Laravel - The PHP Framework For Web Artisans. [online] Laravel.com. Available at: <https://laravel.com/docs/5.1/testing> [Accessed 23 May 2020].

Otwell, T., 2015. Facades - Laravel. [online] Laravel.com. Available at: <https://laravel.com/docs/7.x/facades> [Accessed 22 May 2020].

Otwell, T., 2020. Database: Migrations - Laravel - The PHP Framework For Web Artisans. [online] Laravel.com. Available at: <https://laravel.com/docs/7.x/migrations> [Accessed 23 May 2020].

Otwell, T., 2020. Artisan Console - Laravel - The PHP Framework For Web Artisans. [online] Laravel.com. Available at: <https://laravel.com/docs/7.x/artisan#introduction> [Accessed 22 May 2020].

Otwell, T., 2020. Blade Templates - Laravel - The PHP Framework For Web Artisans. [online] Laravel.com. Available at: <https://laravel.com/docs/7.x/blade#displaying-data> [Accessed 22 May 2020].

Otwell, T., 2020. Middleware - Laravel - The PHP Framework For Web Artisans. [online] Laravel.com. Available at: <https://laravel.com/docs/7.x/middleware#introduction> [Accessed 22 May 2020].

Pereira, S., 2020. University Student Mental Health Survey 2020. [ebook] The Insight Network, p.11. Available at: <https://www.diginbox.com/go/files/Mental%20Health%20Report%202019.pdf>.

Portswigger.net. 2020. What Is CSRF (Cross-Site Request Forgery)? | Web Security Academy. [online] Available at: <https://portswigger.net/web-security/csrf> [Accessed 22 May 2020].

Schwaber, K., 2020. What Is Scrum?. [online] Scrum.org. Available at: <https://www.scrum.org/resources/what-is-scrum> [Accessed 22 May 2020].

Shah, H., 2020. 8 Functional Testing Types Explained With Examples. [online] Insights on Latest Technologies - Simform Blog. Available at: <https://www.simform.com/functional-testing-types/#component> [Accessed 23 May 2020].

Tutorials Point, 2020. Software Design Basics - Tutorialspoint. [online] Tutorialspoint.com. Available at: <https://www.tutorialspoint.com/software_engineering/software_design_basics.htm> [Accessed 22 May 2020].

UK Government, 2019. Data Protection. [online] GOV.UK. Available at: <https://www.gov.uk/data-protection> [Accessed 22 May 2020].

Visual-paradigm.com. 2020. What Is A Class Diagram?. [online] Available at: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/> [Accessed 22 May 2020].

W3schools.com. 2020. SQL Injection. [online] Available at: <https://www.w3schools.com/sql/sql_injection.asp> [Accessed 22 May 2020].

Winter, D., 2020. 9 Benefits Of Test Driven Development. [online] Madetech.com. Available at: <https://www.madetech.com/blog/9-benefits-of-test-driven-development> [Accessed 22 May 2020].

Yodiz Team, 2016. What Is Epic In Agile Methodology (Definition & Template Of Epic). [online] Yodiz Project Management Blog. Available at: <https://www.yodiz.com/blog/what-is-epic-in-agile-methodology-definition-and-template-of-epic/> [Accessed 23 May 2020].

13.Appendices

Appendix 1: User Guide

Manuel Deployment (Ubuntu)

As the project is a web application connected to a database. The following steps must be taken to deploy the application:

1. Install and Setup local MySQL server
2. Install composer
3. Install Laravel
4. Clone repository from https://github.com/Jimmy122999/NationalUniversitySports
5. Rename .env.example to .env and change DB_CONNECTION properties to the local host.
6. Run “php artisan migrate”
7. Run “php artisan serve” to run server
8. Open in web browser
9. Create account
10. Change ‘user_group’ field in database to 1 to make user administrator

Remote Version

There is a remote version of the application hosted at www.downaman.co.uk with the following user accounts:

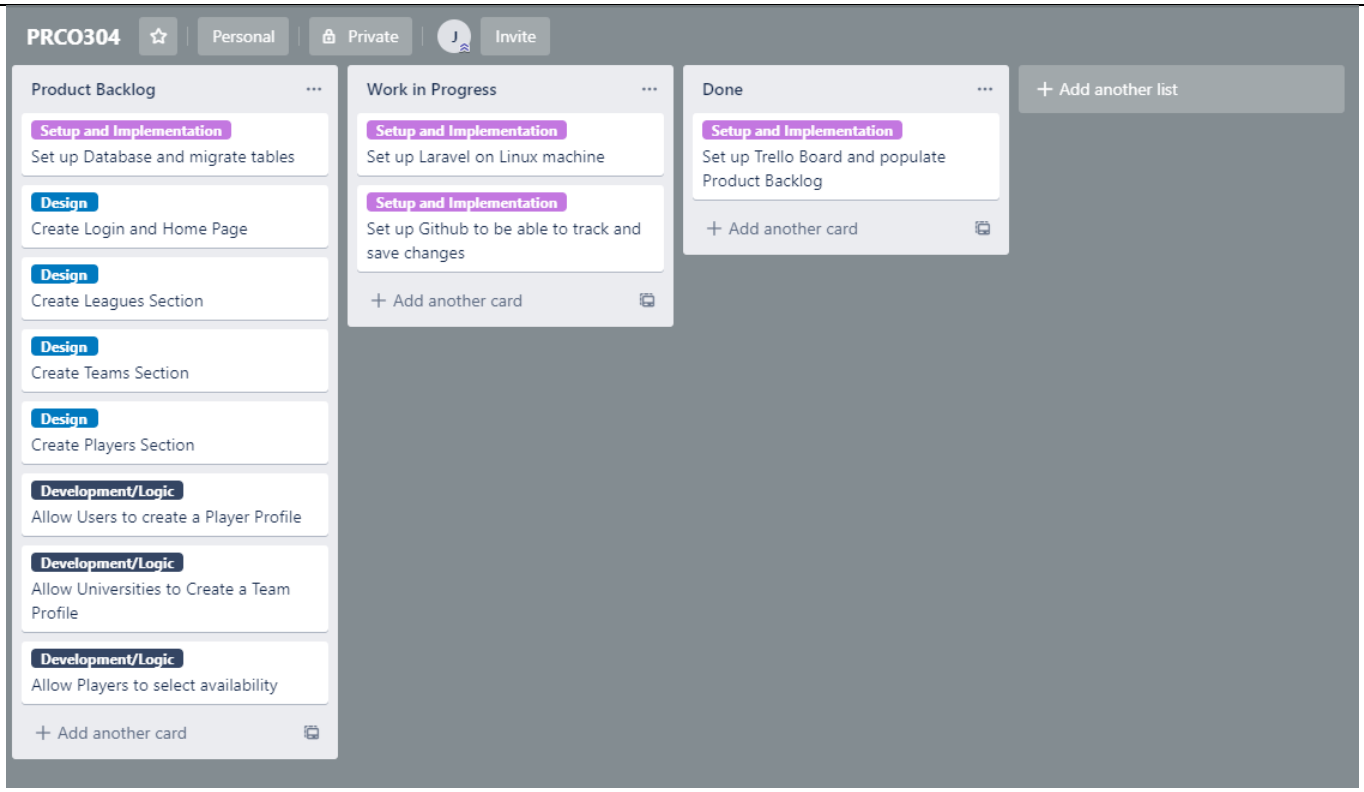
Password recovery does not function on the remote version of the site due to it being developed on a local server.

	Email	Password

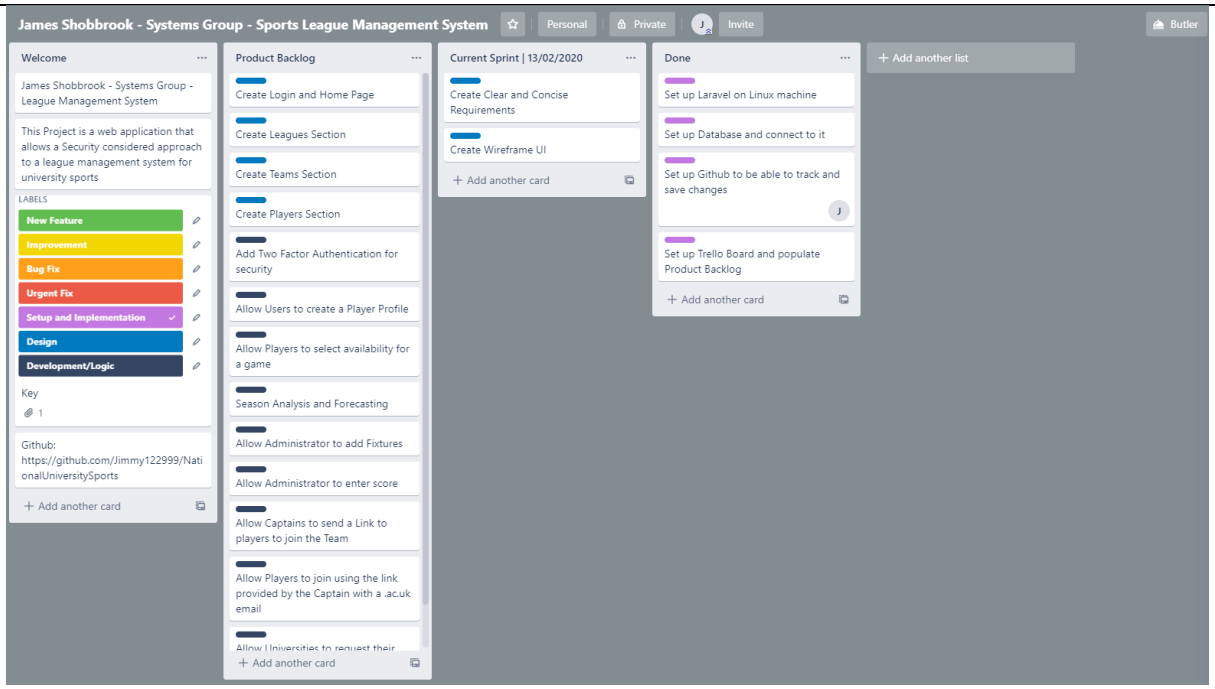
Administrator	Admin1@admin.co.uk	admin123
Captain	Captain1@captain.co.uk	captain123
Player	Player1@player.co.uk	player123

Appendix 2: Trello Boards/User Stories

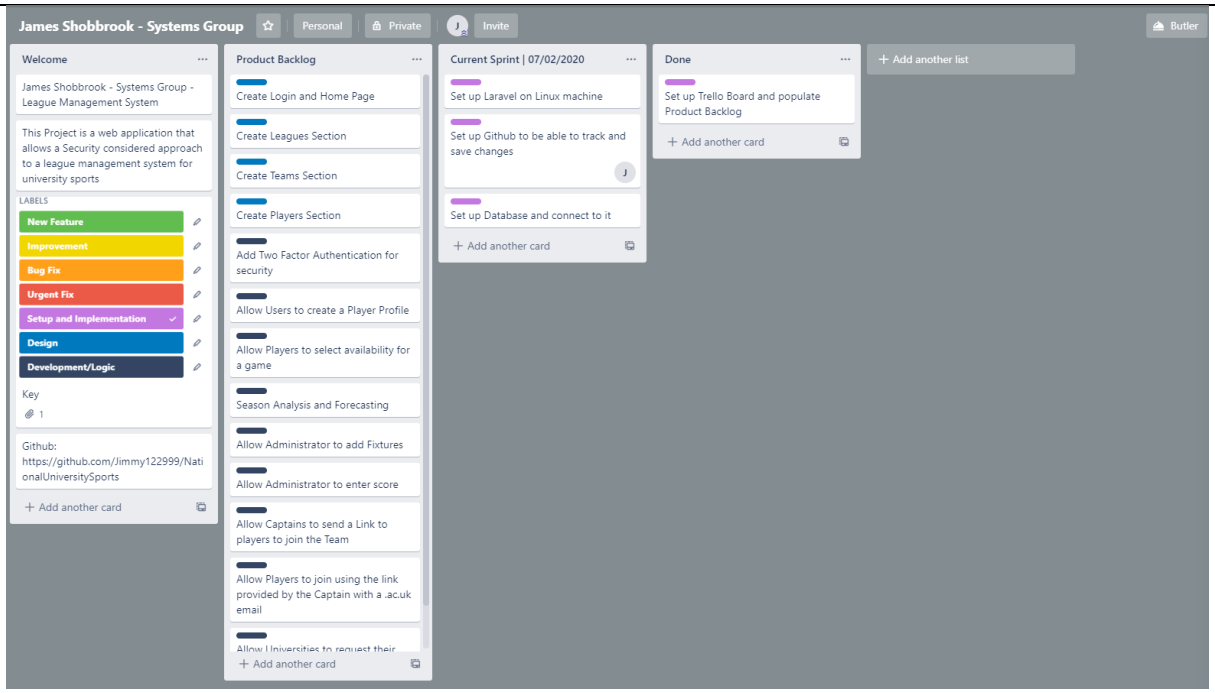
Board 1



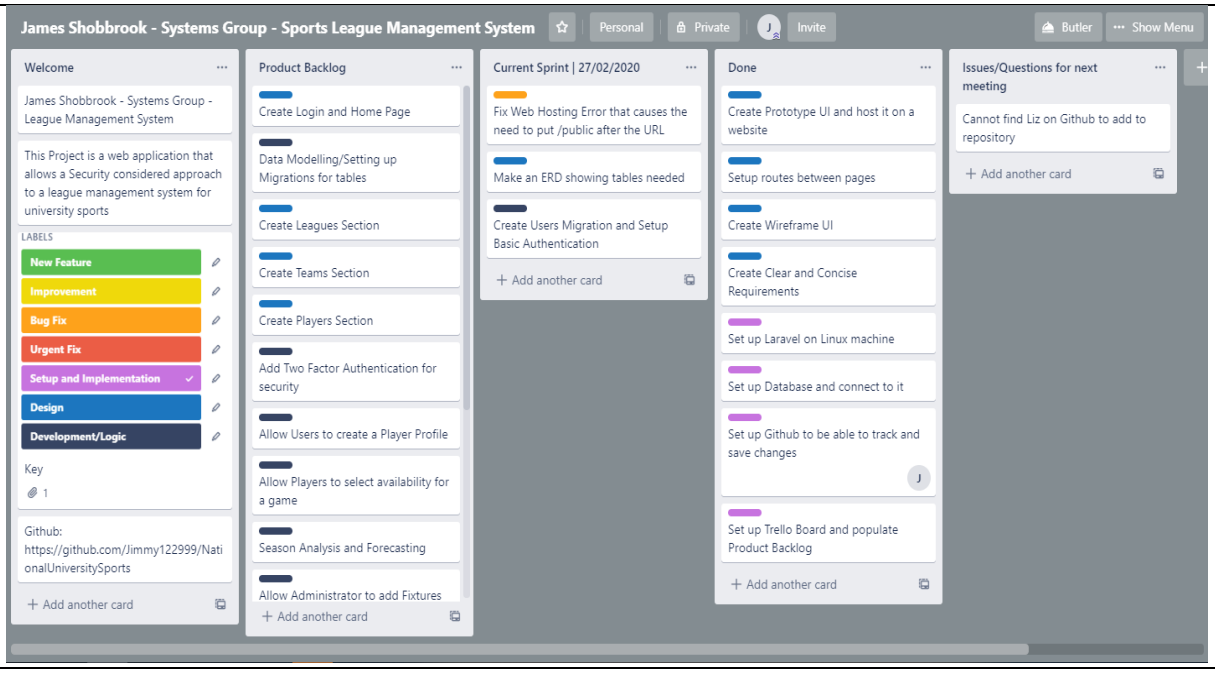
Board 2



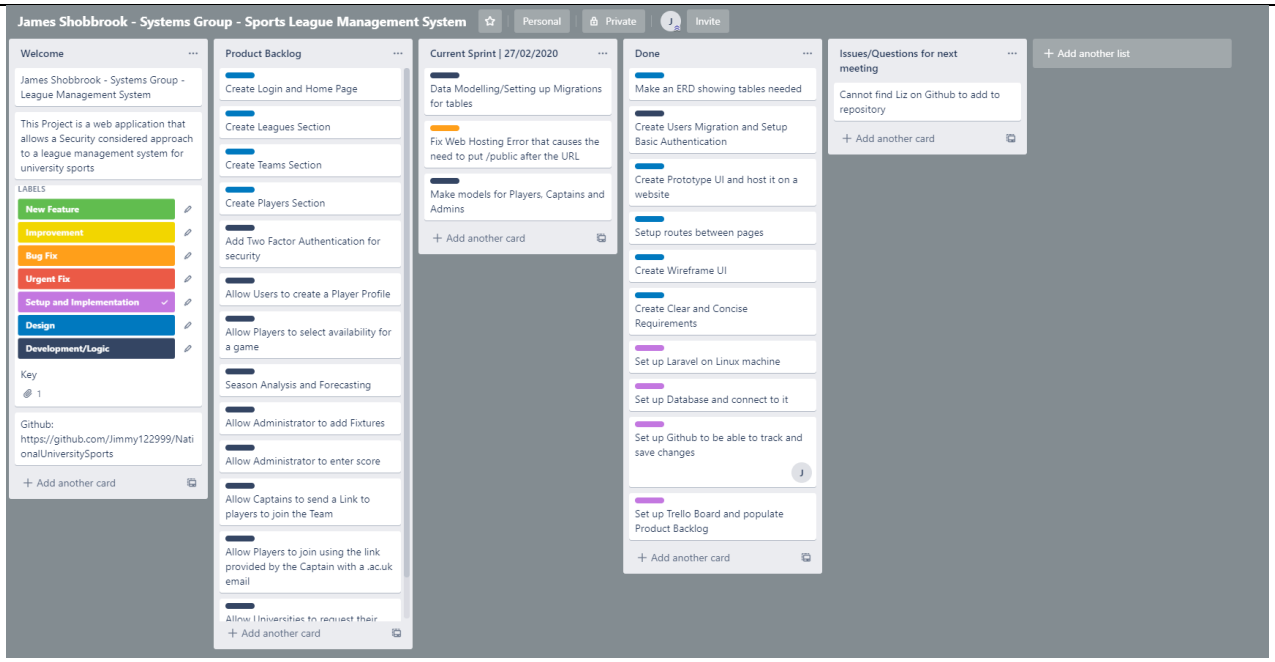
Board 3



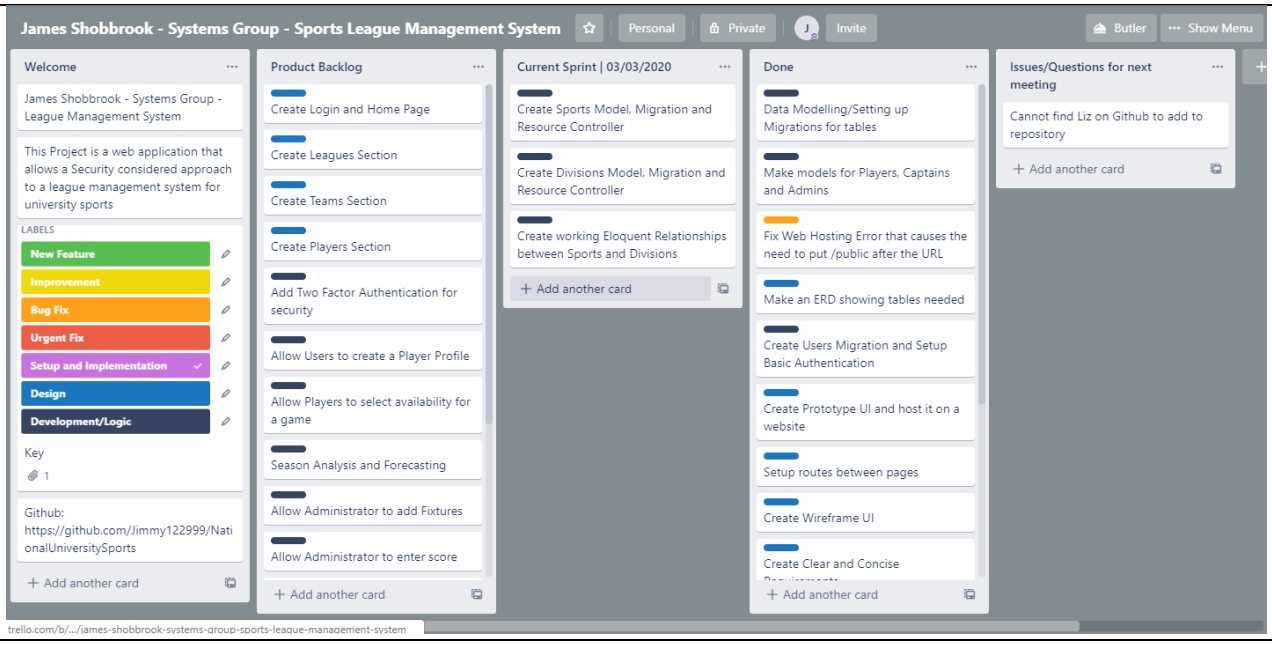
Board 4



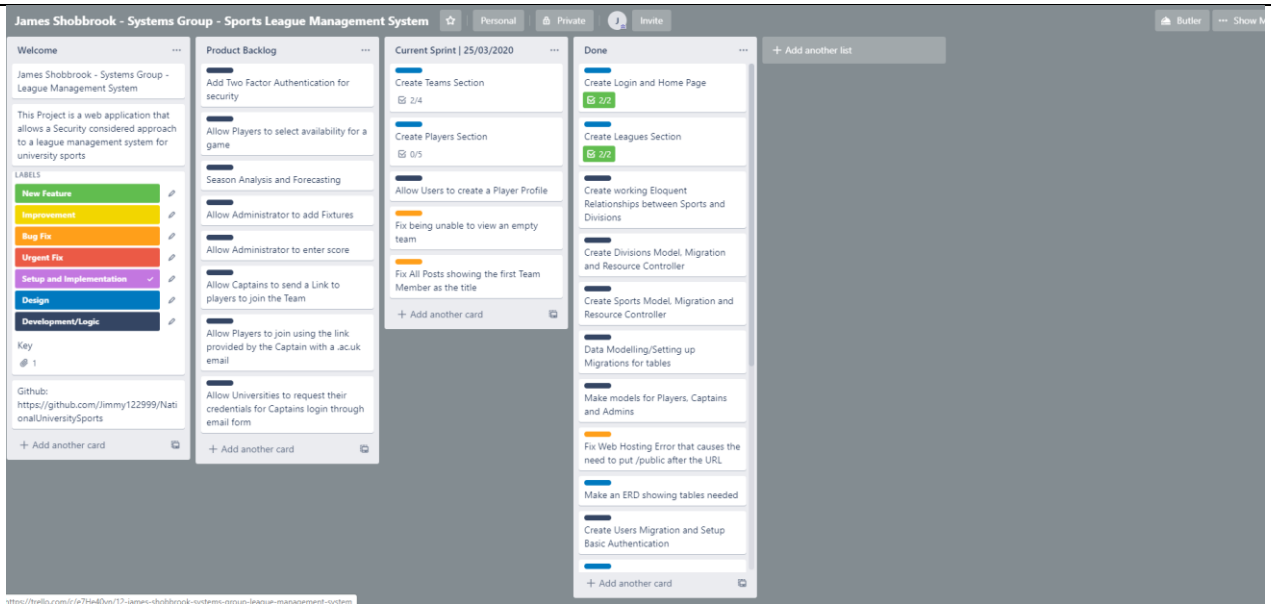
Board 5



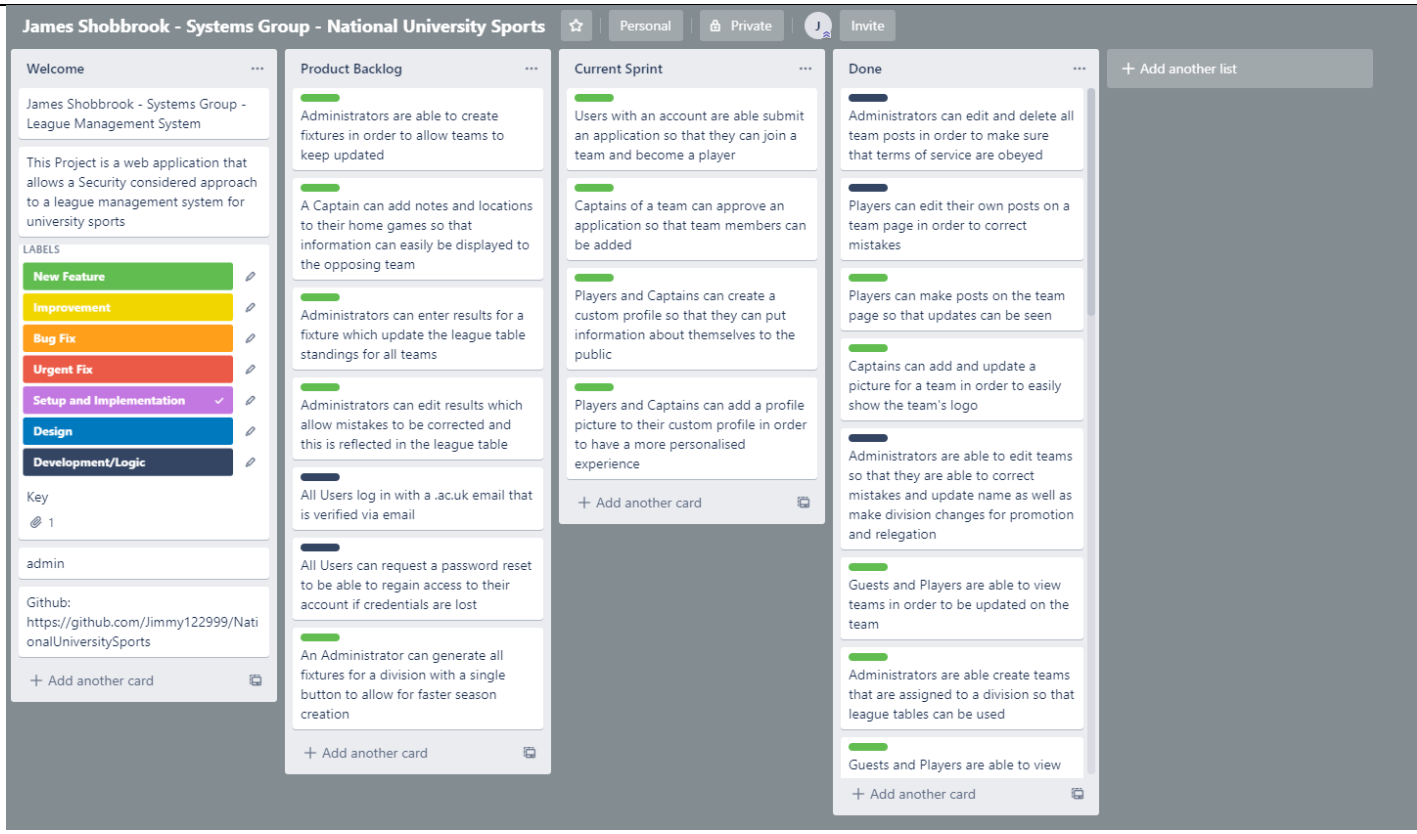
Board 6

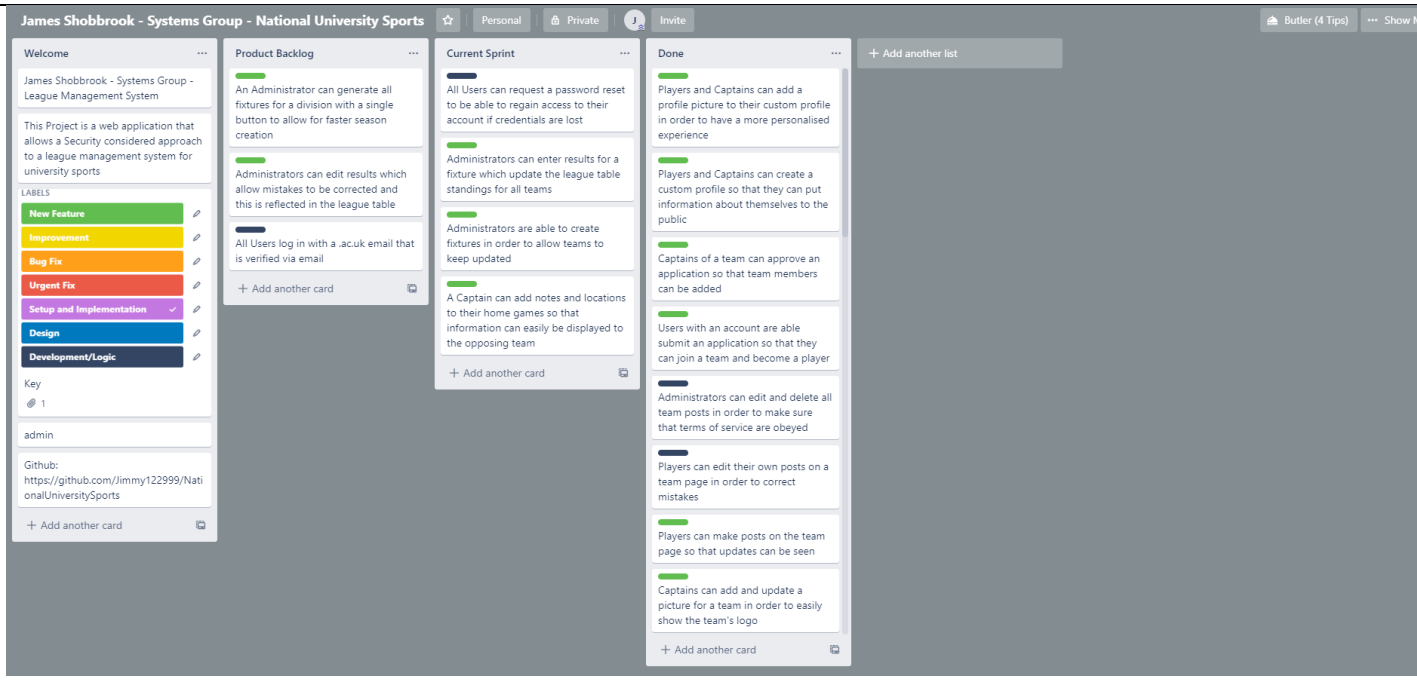
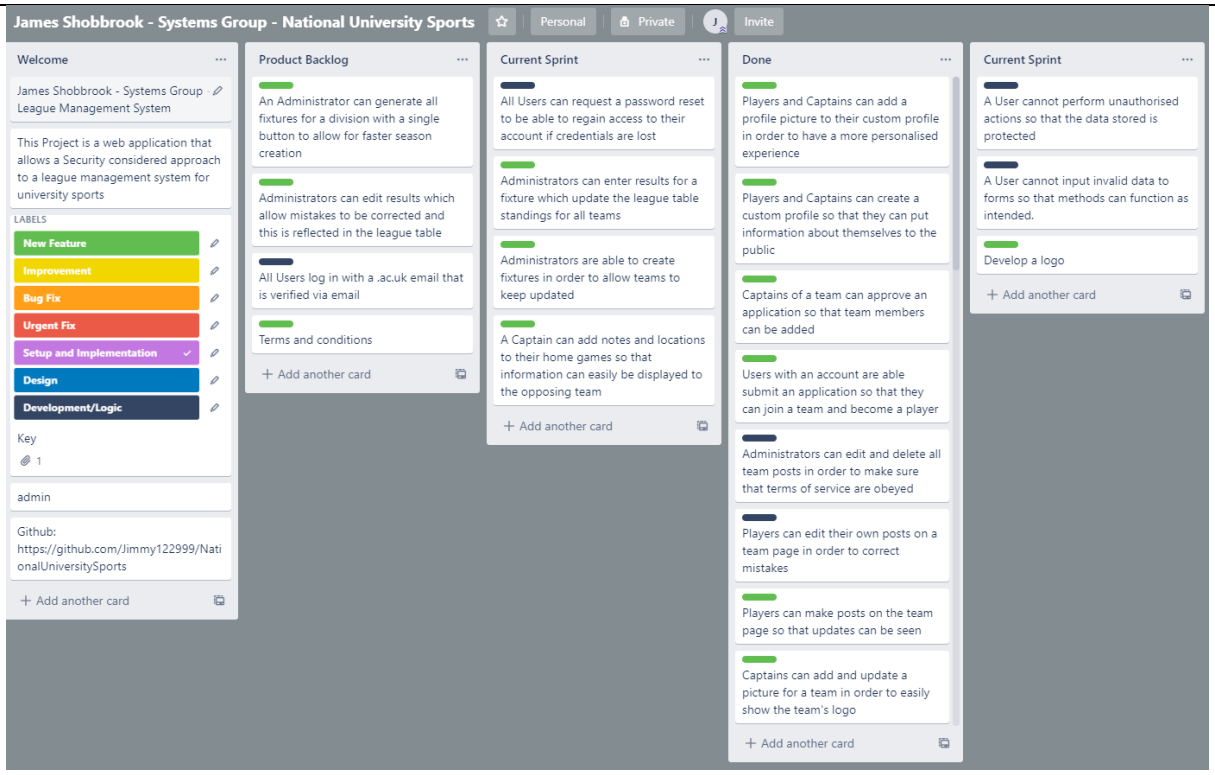


Board 7



Board 8



Board 9	
Board 10	

Appendix 3: Requirements

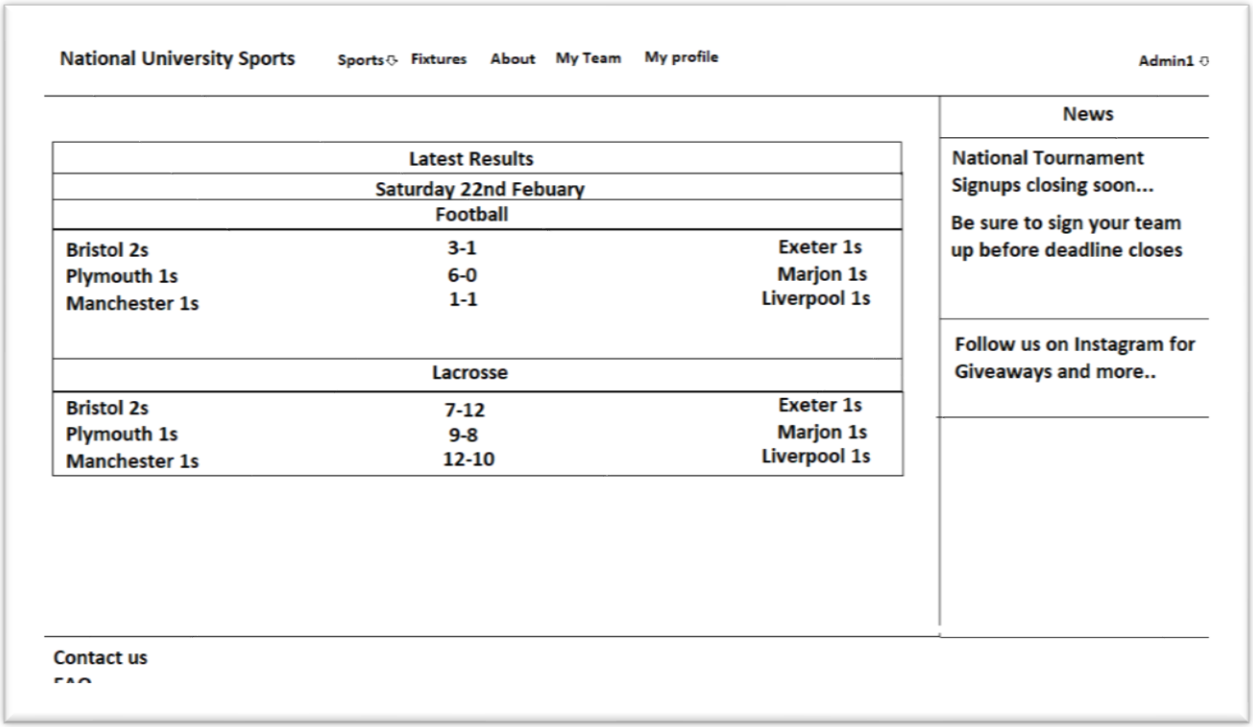
Essentials	
All Users Able to Log into the website with an account with their own credentials	**X**

Three separate user types, **Administrator**, **Team Captain** (Captains) and **Players** each having different roles on the website	**X**
Team Page in which **Captains** can edit information and **Players** can view information	**X**
Administrators can create, read, update, destroy (CRUD) new sports	**X**
Administrators can CRUD divisions that belong to a sport. Divisions contain standings of a league table showing, between teams: Games played, Wins, Draws, Losses, Points. The table should update position based on points.	**X**
Administrators can CRUD teams that belong to a division	**X**
Administrators can CRUD fixtures that belong to a division involving two teams from that division	**X**
Administrators can CRUD fixture results involving two teams. The division league table should update based on result.	**X**
Registered users can apply to join a team	**X**
Captains can approve or deny requests to join their team	**X**
Team Members can CRUD posts on the team page	**X**
Team Members can CRUD a user profile	**X**
Team Members can CRUD posts on their user profile	**X**
All forms have CSRF protection to prevent exploits	**X**
Authorisation measures on administrative actions.	**X**
All passwords encrypted	**X**
Desirables	
Team Members can upload a profile picture	**X**
Captains can upload a team picture on the team page	**X**
Administrators can automatically generate an entire season for a division	**X**
Only a specific team's fixtures displayed on their team page	**X**
Instant messenger platform for **teams** to communicate	

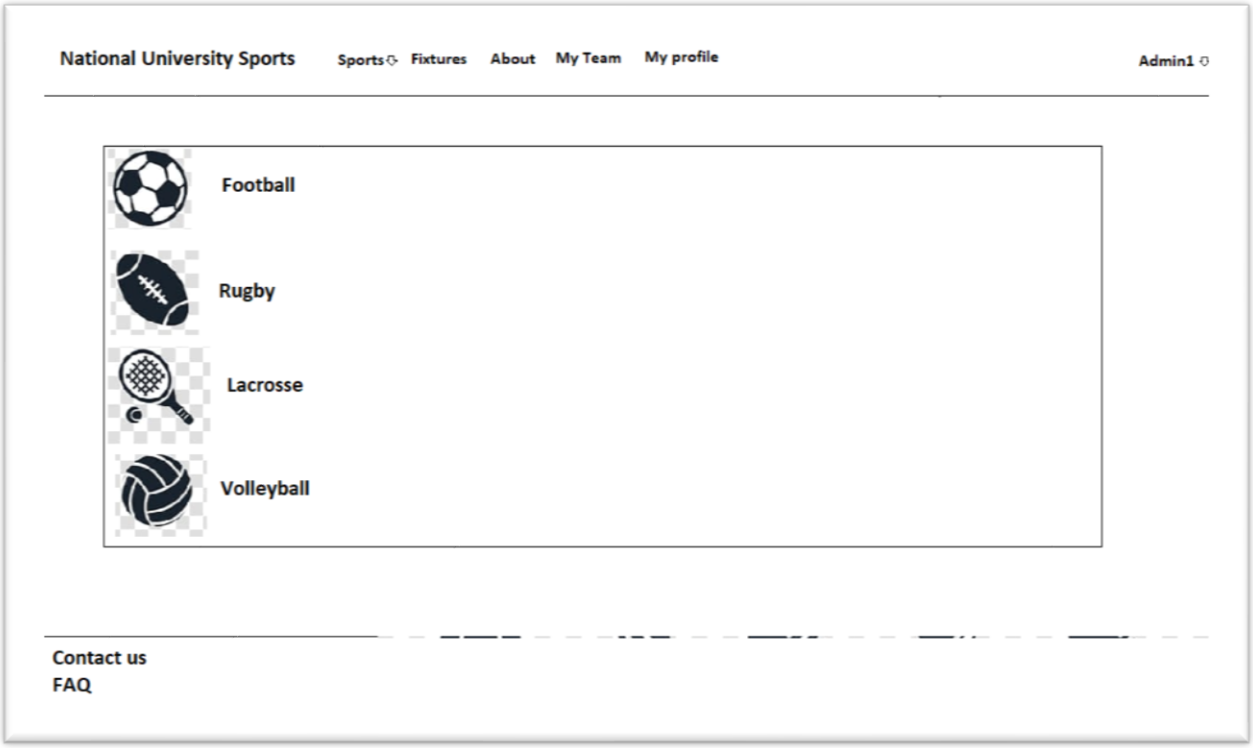
Pitch view of selected **team** with dots and names to represent position	
Captains page where all team captains can communicate.	
Pagination of large amounts of data.	**X**

Appendix 4: GUI Designs

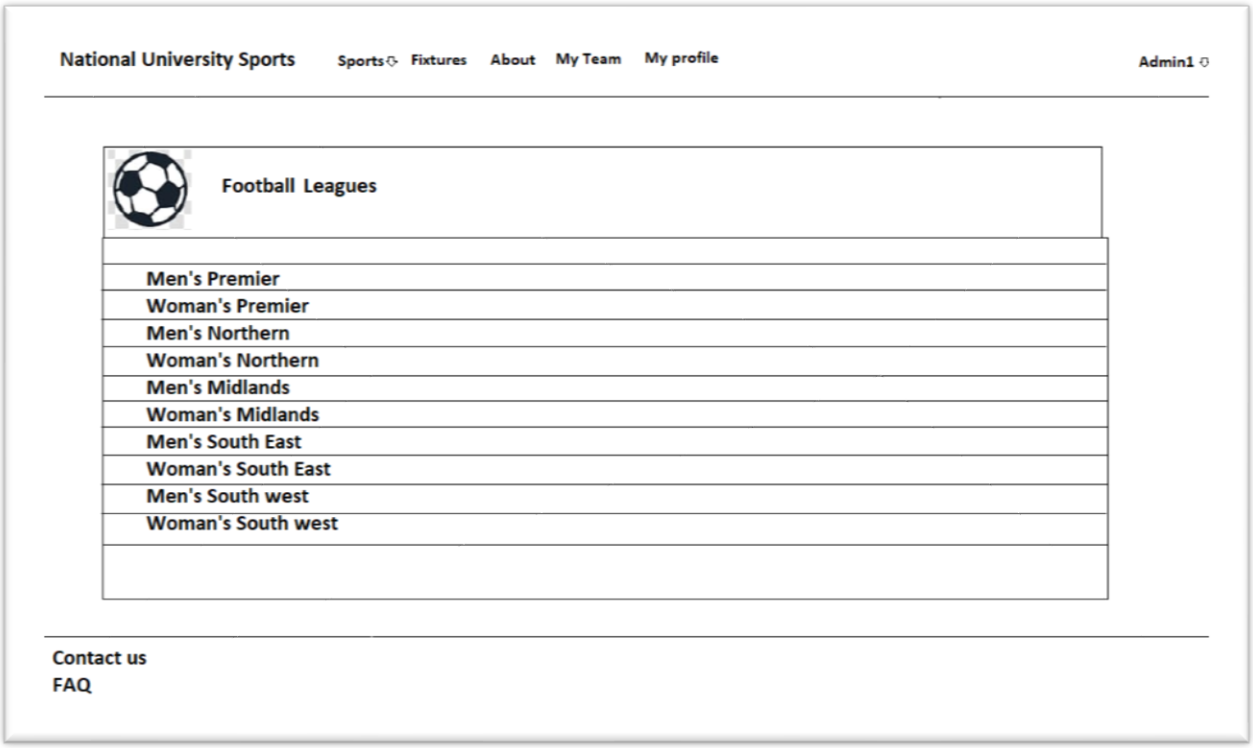
Home Page



Sports Select



Sport Page



Division Page

National University Sports Sports Fixtures About My Team My profile Admin1

Football Mens South West Division 1

Pos	Team	GP	Wins	Draws	Losses	Points
1	Plymouth Mens 1s	2	2	0	0	6
2	Bristol Mens 1s	2	1	1	0	4
3	Exeter Mens 1s	2	1	1	0	4
4	Marjon Mens 1s	2	0	0	2	0

Contact us
FAQ

Fixtures page

National University Sports Sports Fixtures About My Team My profile Admin1

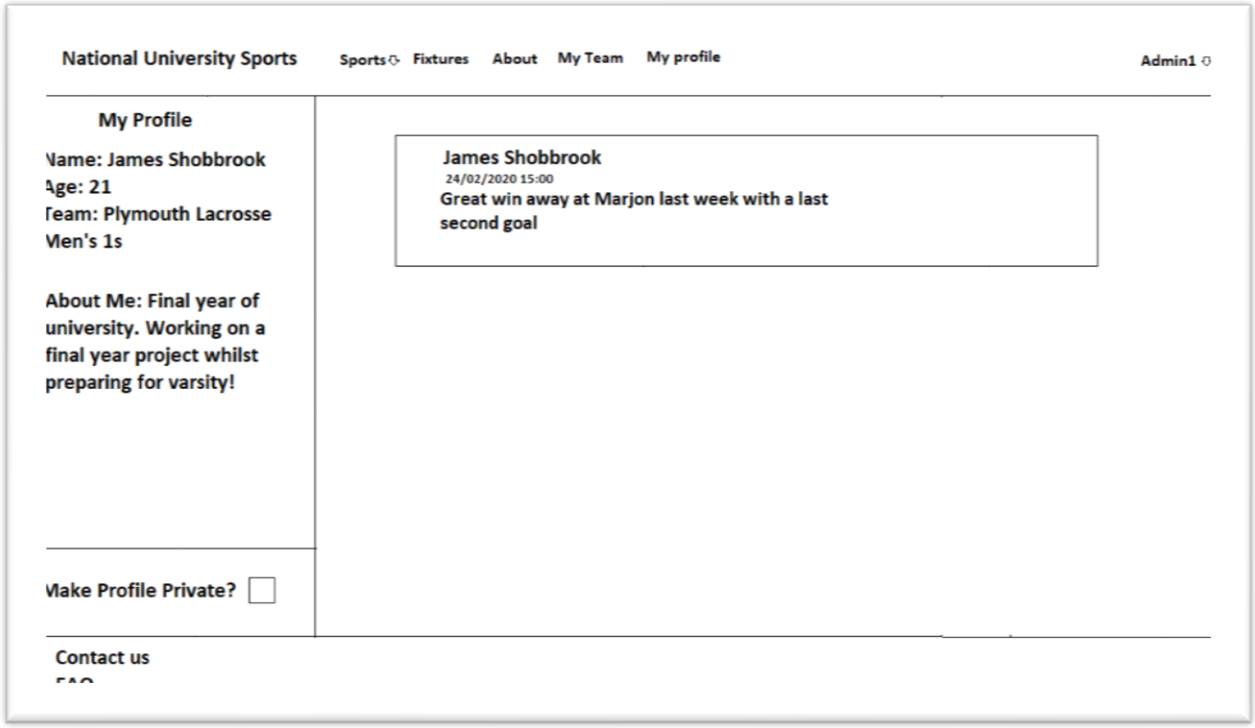
My Fixtures

Wednesday 26th Febuary		
Plymouth Mens 1s	13:00 Brickfields Astroturf Changing Room provided	Southhampton Mens 1s
Available		Unavailable

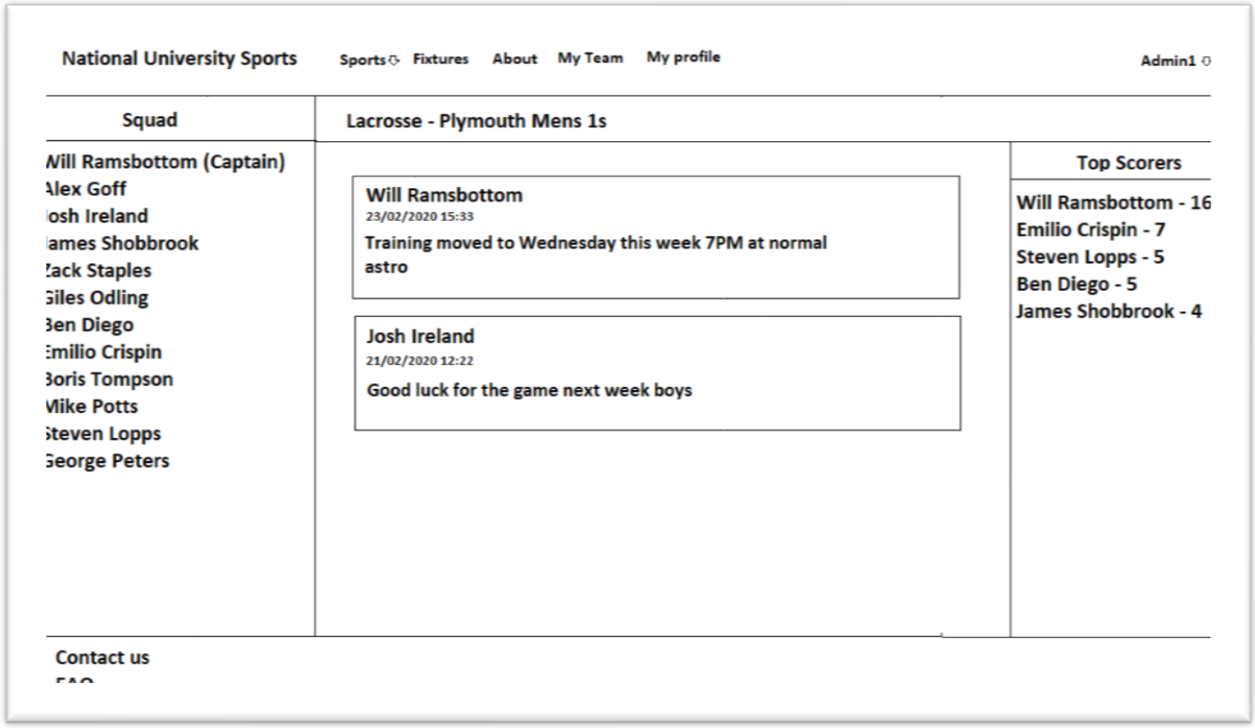
Wednesday 26th Febuary		
Plymouth Mens 1s	13:00 Brickfields Astroturf Changing Room provided	Southhampton Mens 1s
Available		Unavailable

Contact us
FAQ

Profile Page



Team Page



Appendix 5: Black Box Testing

Administrator

Use Case	Test description	Test Values	Expected results	Actual Results	Pass/Fa
Logging in	Entering correct credentials	“admin1@admin.co.uk” “admin123”	Logged in and redirected to admin route	Logged in and redirected to admin route	
	Entering in no credentials		Prompted to enter information into fields	Prompted to enter information into fields	
	Entering incorrect credentials	‘test’ ‘test’	Prompted that credentials are not valid	Prompted that credentials are not valid	
	Changing password	“admin1@admin.co.uk”	Mailtrap.io email sent to email address	Mailtrap.io email sent to email address	
CRUD Sports	Create unique, valid sport	‘Lacrosse’	Sport created and redirected to sport selection	Sport created and redirected to sport selection	
	Create with numerical characters	’23 sport’	Validation error stating only letters and spaces can be used	Validation error stating only letters and spaces can be used	
	Create with Special characters	‘£$Sport’	Validation error stating only letters	Validation error stating only letters	

			and spaces can be used	and spaces can be used	
	Create a duplicate	‘Football’	Validation error stating ‘must be unique’	Validation error stating ‘must be unique’	
	Create too long	‘aaaaaaaaaaaaaaaaaaaaaaa’	Validation error stating ‘must be less than 20 characters’	Validation error stating ‘must be less than 20 characters’	
	Create with empty field		Validation error stating ‘field is required’	Validation error stating ‘field is required’	
	Edit field	‘Rugby’	Record updated and redirected to Sports selection	Record updated and redirected to Sports selection	
	Delete record		Redirected to sports page with record removed	Redirected to sports page with record removed	
CRUD Divisions	Create unique and valid	‘Western Division 1’	Division created and redirected to Division selection	Division created and redirected to Division selection	
	Create too long	‘aaaaaaaaaaaaaaaaaaaaaaaaaaAaaaaaaaaaaaaaaaaaaaaa’	Validation error stating ‘must be less than	Validation error stating ‘must be less than	

			30 characters’	30 characters’	
	Create with empty field		Validation error stating ‘field is required’	Validation error stating ‘field is required’	
	Edit field	‘Western Division 3’	Record updated and redirected to division selection	Record updated and redirected to division selection	
	Delete record		Redirected to division page with record removed	Redirected to division page with record removed	
CRUD Teams	Create valid	‘Plymouth University’ , ‘Football’ , ‘Division 1’	Redirected to created team page	Redirected to created team page	
	Create with numerical characters	‘Plymouth University 1’s’ , ‘Football’ , ‘Division 1’	Redirected to created team page	Redirected to created team page	
	Create with Special characters	‘Plymouth University 1’s’ , ‘Football’ , ‘Division 1’	Redirected to created team page	Redirected to created team page	
	Create too long	‘aaaaaaaaaaaaaaaaaaaaaaaaa Aaaaaaaaaaaaaaaaaaaaaaaaa Aaaaaaaaaaaaaaaaaaaaaaaaa’	Validation error stating ‘must be less than 50 characters’	Validation error stating ‘must be less than 50 characters’	
	Create with empty fields		Validation error stating the fields that must be required	Validation error stating the fields that must be required	

	Edit Name	‘New Team’	Team updated and redirected to team page	Team updated and redirected to team page	
	Edit team’s division	Selecting ‘Western Division 1’ from ajax select box	Division updated and redirected to team page	Division updated and redirected to team page	
	Delete record		Team deleted and redirected to sports select	Team deleted and redirected to sports select	
Administrate Team Page	Clicking accept on an application		Application removed from application page and team member added	Application removed from application page and team member added	
	Clicking deny on an application		Application removed from application page.	Application removed from application page.	
	Uploading picture with a valid picture format		Team page shown with uploaded picture	Team page shown with uploaded picture	
	Uploading picture with an invalid picture format		Validation error stating picture is incorrect format	Validation error stating picture is incorrect format	

	Edit a team post	‘Edited’	Post updated	Post updated	
	Delete a team post		Team page shown with post deleted	Team page shown with post deleted	
Administrate profile Page	Uploading picture with a valid picture format	‘test.jpg’	Profile page shown with uploaded picture	Profile page shown with uploaded picture	
	Uploading picture with an invalid picture format	‘test.sql’	Validation error stating picture is incorrect format	Validation error stating picture is incorrect format	
	Edit a profile post	‘Edited’	Post updated	Post updated	
	Delete a profile post		Profile page shown with post deleted	Profile page shown with post deleted	
CRUD Fixture	Enter valid information	‘Test 1’ ,‘Test 2’ , 12/12/2020 15:00, ‘Freedom fields’	Fixture created and redirected to admin panel	Fixture created and redirected to admin panel	
	Enter empty information		Validation error stating which fields are required.	Validation error stating which fields are required.	
	Edit fixture	‘Test 1’ ,‘Test 2’ , 12/12/2020 13:00, ‘Freedom fields’	Fixture updated and redirected	Fixture updated and redirected	

			to division fixtures	to division fixtures	
	Delete fixture		Redirected to home page and fixture deleted	Redirected to home page and fixture deleted	
CRUD Fixture Result	Enter valid information	‘2’ , ‘2’	Redirected to league table and teams points and games played changed to 1	Redirected to league table and teams points and games played changed to 1	
	Edit fixture result	‘2’ , ‘1’	Redirected to league table with teams and points updated to 3 and 0	Redirected to league table with teams and points updated to 3 and 0	
	Delete fixture result		Redirected to division league table with points and games played correctly reversed	Redirected to division league table with points and games played correctly reversed	
Generate Fixture List	Input Date for season start	12/12/2020	Fixture list correctly generated with 0 clashes and matchdays 2 weeks apart	Fixture list correctly generated with 0 clashes and matchdays 2 weeks apart	

Handle Team Captaincy	Set a registered user as a captain	‘captain1’	User group changed from 3 to 2	User group changed from 3 to 2	
	Remove a captaincy	‘captain1’	User group changed from 2 to 3	User group changed from 2 to 3	

Captain

Use Case	Test description	Test Values	Expected results	Actual Results	Pass/Fail
Logging in	Entering correct credentials	“captain1@captain.co.uk” “captain123”	Logged in and redirected to admin route	Logged in and redirected to admin route	
	Entering in no credentials		Prompted to enter information into fields	Prompted to enter information into fields	
	Entering incorrect credentials	‘test’ ‘test’	Prompted that credentials are not valid	Prompted that credentials are not valid	
View Sports	Clicking sports button		Redirected to sports page	Redirected to sports page	
	Clicking a sport		Directed to the divisions of that sport	Directed to the divisions of that sport	
View Divisions	Clicking a division		Division listed with league table, fixtures and results	Division listed with league table, fixtures and results	
View Team	Clicking on a team name		Directed to team page	Directed to team page	

Manage Team	View applications button		Directed to pending applications	Directed to pending applications	
	Accept an application		Member added to team and application deleted	Member added to team and application deleted	
	Deny an application		Application deleted and removed	Application deleted and removed	
CRUD Team post	Entering a valid post	‘Fantastic display from the team’	Redirected to team page with post added	Redirected to team page with post added	
	Entering an empty post		Validation error stating field is required	Validation error stating field is required	
	Edit own post	‘New edit’	Redirected to team page with post edited	Redirected to team page with post edited	
	Click delete button		Redirected to team page with post deleted	Redirected to team page with post deleted	
CRUD Profile	Entering valid information	‘Defender’ ‘New player’	Redirected to created profile	Redirected to created profile	
	Entering empty information		Validation error stating fields required	Validation error stating fields required	
	Uploading picture with a valid picture format		Profile page shown with uploaded picture	Profile page shown with uploaded picture	
	Uploading picture with		Validation error	Validation error	

	a invalid picture format		stating picture is incorrect format	stating picture is incorrect format	
	Editing information	‘Goalkeeper’ , ‘New bio’	Redirected to profile	Redirected to profile	
	Deleting profile		Redirected to home page	Redirected to home page	
CRUD Profile post	Entering a valid post	‘Great game’	Redirected to profile page with post added	Redirected to profile page with post added	
	Entering an empty post		Validation error stating field is required	Validation error stating field is required	
	Edit own post	‘New edit’	Redirected to profile page with post edited	Redirected to profile page with post edited	
	Click delete button		Redirected to profile page with post deleted	Redirected to profile page with post deleted	
Leave team	Clicking leave team button		Removed from team and team abilities removed	Removed from team and team abilities removed	
Manage Fixtures	Entering valid information to a home fixture	‘Freedom fields’	Fixture information updated and redirected to divisions fixtures	Fixture information updated and redirected to divisions fixtures	

Player/Authenticated User

Use Case	Test description	Test Values	Expected results	Actual Results	Pass/Fail
Logging in	Entering correct credentials	“player1@player.co.uk” “player123”	Logged in and redirected to admin route	Logged in and redirected to home page	
	Entering in no credentials		Prompted to enter information into fields	Prompted to enter information into fields	
	Entering incorrect credentials	‘test’ ‘test’	Prompted that credentials are not valid	Prompted that credentials are not valid	
View Sports	Clicking sports button		Redirected to sports page	Redirected to sports page	
	Clicking a sport		Directed to the divisions of that sport	Directed to the divisions of that sport	
View Divisions	Clicking a division		Division listed with league table, fixtures and results	Division listed with league table, fixtures and results	
View Team	Clicking on a team name		Directed to team page	Directed to team page	
Applying for a team a team	Entering Display name	‘Rodger Rabbit’	Redirected to team page	Redirected to team page	
CRUD Team post	Entering a valid post	‘Fantastic display from the team’	Redirected to team page with post added	Redirected to team page with post added	

	Entering an empty post		Validation error stating field is required	Validation error stating field is required	
	Edit own post	‘New edit’	Redirected to team page with post edited	Redirected to team page with post edited	
	Click delete button		Redirected to team page with post deleted	Redirected to team page with post deleted	
CRUD Profile	Entering valid information	‘Defender’ ‘New player’	Redirected to created profile	Redirected to created profile	
	Entering empty information		Validation error stating fields required	Validation error stating fields required	
	Uploading picture with a valid picture format		Profile page shown with uploaded picture	Profile page shown with uploaded picture	
	Uploading picture with a invalid picture format		Validation error stating picture is incorrect format	Validation error stating picture is incorrect format	
	Editing information	‘Goalkeeper’ , ‘New bio’	Redirected to profile	Redirected to profile	
	Deleting profile		Redirected to home page	Redirected to home page	
CRUD Profile post	Entering a valid post	‘Great game’	Redirected to profile page with post added	Redirected to profile page with post added	

	Entering an empty post		Validation error stating field is required	Validation error stating field is required	
	Edit own post	‘New edit’	Redirected to profile page with post edited	Redirected to profile page with post edited	
	Click delete button		Redirected to profile page with post deleted	Redirected to profile page with post deleted	
Leave team	Clicking leave team button		Removed from team and team abilities removed	Removed from team and team abilities removed	

Guest/Unauthenticated User

Use Case	**Test description**	**Test Values**	**Expected results**	**Actual Results**	**Pass/Fail**
Registering account	Enter valid information	‘Tester’, test@test.com’ , ‘password123!’ , ‘password123!’	Account created and redirected to logged in dashboard	Account created and redirected to logged in dashboard	
	Entering a weak password	‘Tester’, test@test.com’ , ‘pas!’ , ‘pa!’	Validation error asking for a stronger password	Validation error asking for a stronger password	
	Entering a duplicate email	‘Tester’, admin1@admin.co.uk’ , ‘password123!’ , ‘password123!’	Validation error saying email is already taken	Validation error saying email is already taken	

View Sports	Clicking sports button		Redirected to sports page	Redirected to sports page	
	Clicking a sport		Directed to the divisions of that sport	Directed to the divisions of that sport	
View Divisions	Clicking a division		Division listed with league table, fixtures and results	Division listed with league table, fixtures and results	
View Team	Clicking on a team name		Directed to team page	Directed to team page	
View Profile	Clicking on a profile name		Directed to profile page	Directed to profile page	
View Fixtures	Clicking on Upcoming fixtures for a division		Directed to a division's fixtures page	Directed to a division's fixtures page	
View Results	Clicking on Upcoming results for a division		Directed to a division's results page	Directed to a division's results page	