

National College of Ireland  
BSc in Computing  
2015/2016

Ian Costello  
X12112348  
lancostello120@gmail.com



# Sport Leagues

Technical Report



National  
College *of*  
Ireland

## **Declaration Cover Sheet for Project Submission**

### **SECTION 1 Student to complete**

**Name:**

**Student ID:**

**Supervisor:**

### **SECTION 2 Confirmation of Authorship**

**The acceptance of your work is subject to your signature on the following declaration:**

**I confirm that I have read the College statement on plagiarism (summarised overleaf and printed in full in the Student Handbook) and that the work I have submitted for assessment is entirely my own work.**

**Signature:**\_\_\_\_\_

**Date:**\_\_\_\_\_

**NB. If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the College's Disciplinary Committee. Should the Committee be satisfied that plagiarism has occurred this is likely to lead to your failing the module and possibly to your being suspended or expelled from college.**

**Complete the sections above and attach it to the front of one of the copies of your assignment,**

**What constitutes plagiarism or cheating?**

**The following is extracted from the college's formal statement on plagiarism as quoted in the Student Handbooks. References to "assignments" should be taken to include any piece of work submitted for assessment.**

**Paraphrasing refers to taking the ideas, words or work of another, putting it into your own words and crediting the source. This is acceptable academic practice provided you ensure that credit is given to the author. Plagiarism refers to copying the ideas and work of another and misrepresenting it as your own. This is completely unacceptable and is prohibited in all academic institutions. It is a serious offence and may result in a fail grade and/or disciplinary action. All sources that you use in your writing must be acknowledged and included in the reference or bibliography section. If a particular piece of writing proves difficult to paraphrase, or you want to include it in its original form, it must be enclosed in quotation marks**

**and credit given to the author.**

**When referring to the work of another author within the text of your project you must give the author's surname and the date the work was published. Full details for each source must then be given in the bibliography at the end of the project**

### **Penalties for Plagiarism**

**If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the college's Disciplinary Committee. Where the Disciplinary Committee makes a finding that there has been plagiarism, the Disciplinary Committee may recommend**

- **that a student's marks shall be reduced**

- that the student be deemed not to have passed the assignment
- that other forms of assessment undertaken in that academic year by the same student be declared void
- that other examinations sat by the same student at the same sitting be declared void

Further penalties are also possible including

- suspending a student college for a specified time,
- expelling a student from college,
- prohibiting a student from sitting any examination or assessment.,
- the imposition of a fine and
- the requirement that a student to attend additional or other lectures or courses or undertake additional academic work.

# Table of Contents

Executive Summary .....	8
1 Introduction.....	9
1.1 Project research.....	9
1.2 Background.....	10
1.3 Aims.....	10
1.4 Technologies overview .....	11
1.4.1 Angular js .....	11
1.4.2 Node js .....	12
1.4.3 MongoDB.....	12
1.4.4 Ionic mobile development.....	12
2 System.....	13
2.1 Requirements .....	13
2.2 Functional requirements .....	13
2.2.1 Use Case Diagram .....	15
2.2.2 Requirement login .....	15
2.2.3 Requirement registration .....	17
2.2.4 Requirement Leagues .....	18
2.2.5 Requirement Profile.....	20
2.2.6 Requirement Picks .....	22
2.2.7 Requirement Search .....	23
2.3 Non-Functional Requirements .....	24
2.3.1 Performance/Response time requirement.....	24
2.3.2 Availability requirement .....	25
2.3.3 Recover requirement.....	25
2.3.4 Robustness requirement .....	25
2.3.5 Security requirement .....	25
2.3.6 Reliability requirement.....	25
2.3.7 Extendibility requirement.....	25
2.3.8 Reusability requirement.....	25
2.3.9 Data requirements.....	25

2.3.10	User requirements.....	26
2.3.11	Environmental requirements .....	26
2.3.12	Usability requirements.....	26
2.4	Design and Architecture.....	26
2.5	Implementation .....	27
2.5.1	NPM .....	27
2.5.2	UI.....	28
2.5.3	Server.....	28
2.5.4	Data and Mongoose (Mongodb ODM) .....	29
2.5.5	<i>Users selections</i> .....	30
2.5.6	Scores and Calculations.....	31
2.6	Testing.....	33
2.6.1	Usability testing .....	33
2.6.2	UNIT Testing .....	33
2.6.3	UAT Testing .....	33
2.7	Graphical User Interface (GUI) Layout.....	34
2.7.1	Login .....	34
2.7.2	Register.....	34
2.7.3	Leagues page .....	35
2.7.4	Create League .....	35
2.7.5	League .....	36
2.8	Customer testing.....	37
2.9	Encountered problems and solutions.....	37
	Conclusions .....	40
3	Further development or research.....	41
4	References .....	42
5	Appendix.....	43
5.1	Project Proposal .....	43
5.2	Project Plan .....	47
	47	
5.3	Monthly Journals.....	48
5.4	UAT testing results .....	59

5.5	Usability testing results .....	61
-----	---------------------------------	----

## **Executive Summary**

The online gambling industry I seeing massive growth due to the increase in technology as well as younger users growing up with the new mobile technology, there seems to be a market for predicting outcomes for sport matches, there are similar applications running sport competitions likes Sky's super six but does not let the users build their own leagues and only users correct score.

Sport Leagues will allow admins more freedom for building leagues and showing visual stats of users within the league.

The current game as it stands is run in excel sheets by and the client wishes to automate the process, he wishes to create not just one league but many and also allow other admins build their own leagues.

The application has potential to sell its data to other sites so they can host there league and leave the business logic up to Sport Leagues application.

A second way of monetising would be through a gambling license and taking a cut of the pot from each competition.

The project will be built in web storm inteji IDE, the main language in use is JavaScript, using Angular, node and mongo DB for the web app and server and ionic mobile app for android phones.

The frameworks I will be using are Express js for the server, Ionic for mobile development and Angular js for both the web and mobile clients.

The application will be hosted using either digital ocean or Heroku



# 1 Introduction

Sport Leagues will be both web and Mobile based.

Using Android for Mobile and Angular for the Web client, Node js for backend development

The aim for this application is for sport lovers who enjoy holding their own competitions based around their favourite sport.

## 1.1 *Project research*

For research on doing the project I looked into the gambling industries examples of online games and how the layout the fixtures and scoring as well as results of football match's.

While I liked there format I didn't see a decent mobile version for this type of application except for fantasy football from sky sports which is a game in a different format.

Fantasy football dominates the app stores so offering an alternative game to users can allow sport leagues to fill this gap for mobile gambling within Ireland.

Sky don't offer an Api for football data, so I used foot-api.com which scrapes the data for me and allows me to access the fixtures and results of match's.

The data is relatively cheap 15 euro per month, compared to high end sports data companies that cost 8000 euro per month.

The data supplies the fixtures for the next seven days and offers live score updates

I can use this to show the user the matches let them choose the team they want to win and then after the results come in from those matches I can then calculate their scores and present it back to the user in a table.

Also I researched the use of node js and how companies such LinkedIn use node js for backend mobile services.

Also a lot of gambling applications are not available on the play store due to gambling laws, The game in the current form is popular but is been operated through friends on whatsapp and storing the data on excel sheets. The current game has meet its limits on the amount of people within a competition (60) and has become to much work for my client to store 60 users selections and tally up the scores and sending people the table each week.

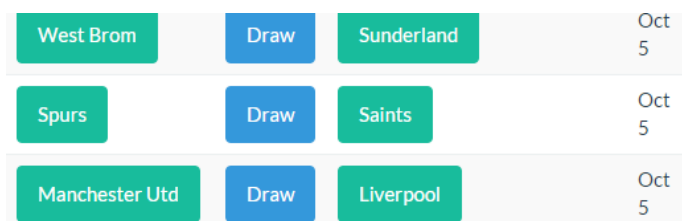
The ideal would target users who submit the selection on mobile and allow for admin section on a web base application.

## 1.2 Background

The plan is to build a web and mobile application using restful services, this application will be design so it can be used with other services and possible developed into a working product for the gambling industry. The online gambling industry is growing business and I believe automating games people play with other like last man standing and so on, the user can have a better experience. Also adding the social nature with games is a must which is rarely seen in the likes of paddy power or Ladbrokes.

The project is s called sport league, a very generic name for a project but I don't want to infringe on other companies' names.

The idea is to do with premiership football, each week there is 10 football matches, people can choose the outcome of the match by either choosing a home win Draw or away win. The image below is an example.



West Brom	Draw	Sunderland	Oct 5
Spurs	Draw	Saints	Oct 5
Manchester Utd	Draw	Liverpool	Oct 5

The image is mine, it's a screen shot from my site in development.

The game it self is currently been played out over

The users will be allocated points for each prediction and they shall be presented the results in a very user friendly UI in web and Mobile, the results will be shown in the table from first to last.

The user will be enter into a league table with other users, and after a set period of weeks a winner will determined by coming 1<sup>st</sup> and top of the league when the league ends.

There will also be an admin area so the admin can control who can join the league, change the league name and allow for adding new features in the future.

The System will calculate the scores for the users

## 1.3 Aims

The aim of this application is bring a new option for sport fans to engage in the sport they love by predicting the outcomes, many fans would travel to the local bookies each weekend placing small sums of money for the entertainment of predicting the winners of the premier league soccer matches, the aim is offer these people another option to be involved in predicting but against other users instead of the bookies. The app is targeted mainly at males between 18 and over, the UI must be simple as the older the users are less tech savvy and will only wish to submit their selections and get there results , either it be using a laptop or using a mobile phone.

Using Ionic mobile development I aim to build sport leagues of android and then further development outside of the 4<sup>th</sup> year time frame to IOS.

The mobile app in the beginning will have limited features compared to the web app but will have the most essential features to optimise the user experience.

For mobile,

The users will be able to login, register, view league fixtures, stats and leagues tables of both the competition (premier league) and their own league table, submit their selections and search and join leagues, the table will be updated after all the results are in from the weekend depending on the dates of the fixtures output by the competition (premier league).

All the services for the mobile app will come from the Sport leagues server using http sending JSON objects and will be authenticated by JSON web tokens. JSON web tokens add a better layer of security for identifying who the user is.

For Web,

The user will be able to login, search, view leagues, stats, league tables, edit their name and the team they support, the admin will be able to edit the league view members and allow members to join if the league is private, the server will use RESTful endpoints and will have different endpoints from mobile as it uses JSON web tokens, while it would be better for security it's not a necessity due to the use of cookies.

## **1.4 Technologies overview**

The application were built using web storm Intellij IDE for development. The technologies involved are:

The main language in use is JavaScript, JavaScript is dominating web development and makes sense to complete the full stack in the same language for rapid development.

The data will be exchanged over RESTful endpoints using JSON.

The site will be hosted either on Digital cloud hosting or using Heroku cloud hosting.

The web application will have scripts running to get the football data daily into a database and then used by the web app and the mobile application

### **1.4.1 Angular js**

Both the web app and mobile application will be using JavaScript.

The UI is built using Angular js, Angular is a JavaScript framework which is both used in the web app and in the ionic mobile app. Angular js will be used on the client end for receiving data from the server.

### **1.4.2 Node js**

The web server is built using node js framework express, the frame work helps with building the architecture of the application for restful web services so I can connect the front end client in easily.

Node it self is very powerful once used correctly and is a module base , meaning adding to node is simple and building multiple node apps helps with rapid development

### **1.4.3 MongoDB**

The database is a extremely important part of the I choose mongo because it natural stores data in JSON, instead of using tables and columns and rows it uses documents which can be stored as arrays , allowing parent and child array in a collection, for example storing username and password and in a sub document contain the users friends.

### **1.4.4 Ionic mobile development**

Ionic is a mobile app framework, its used for rapid development of applications and has examples of apps out there competing with native applications create in Java , one of the main advantages is that it uses angular js which I am already uses and should be able create an application in relative short time compared to making in its native counterpart.

Ionic can compile apps to Android or IOS.

Only drawback is that it can be slower than the natives apps but there is ways to optimise ionic apps , especially in my case and I will not be using many images across the site just data.

## **2 System**

### **2.1 Requirements**

I gathered the requirements with talking to my client who runs the league and users of the league that he is running, the outcome of the requirements the league users aged 18 and above gave me a good insight in the core essentials that make the game as it stands entertaining and trying to increase the user experience and reduce the work load from the user and the admin running the competition .

This led to the requirements of the application.

### **2.2 Functional requirements**

#### Login

- Token based(mobile)
- Access to site

#### Registration

- Access to site

#### Leagues

- Calculate league scores
- Allow users join leagues
- Create Leagues
- Search Leagues

#### Profile

- User can view profile
- Edit profile
- Do not allow duplicate user names or email accounts
- Select team

### Selecting picks

- User must be able select picks and edit picks update database
- Calculate league stats

### Admin

- Allow users to join private league with permission from admin
- Edit league name ,description and if its private and public

### Live scores

- Select date

### Log out

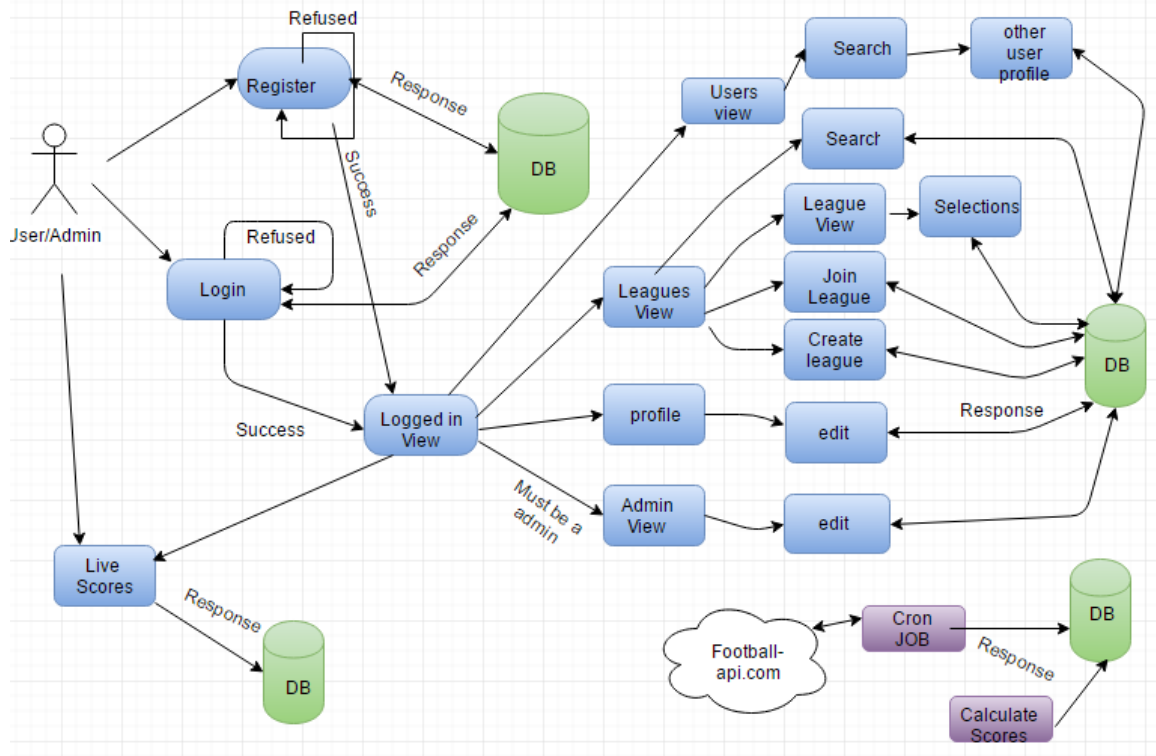
- Destroy session
- Redirect user to login

### Database

- Stateless connection

## 2.2.1 Use Case Diagram

This Use Case Diagram provides an overview of all functional requirements.



## 2.2.2 Requirement login

### *Use Case 1.0*

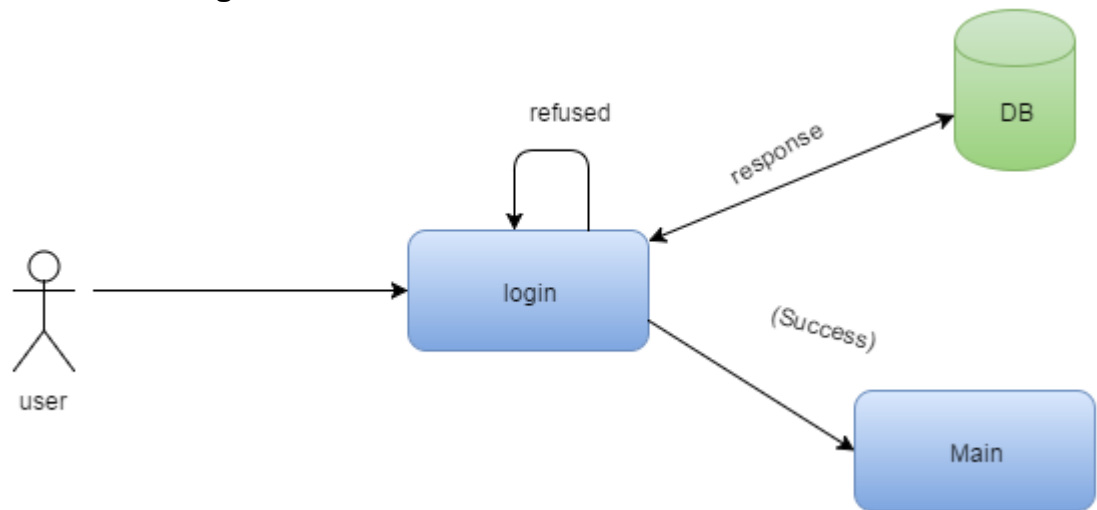
#### Scope

Login

#### Description

Login

## Use Case Diagram



## Flow Description

### Precondition

System is idle

### Activation

User clicks login

### Main flow

1. User clicks login
2. User enters username and password A1
3. User enters main page

### Alternate flow

1. User clicks login
2. User enters username and password
3. User credentials wrong or user does not exist
4. User returned to login page



### Termination

- User leaves application

### Post condition

The system goes into a wait state

## 2.2.3 Requirement registration

### 2.2.3.1 Description & Priority

Registration for user's login and refusing duplicate accounts

### 2.2.3.2 Use Case

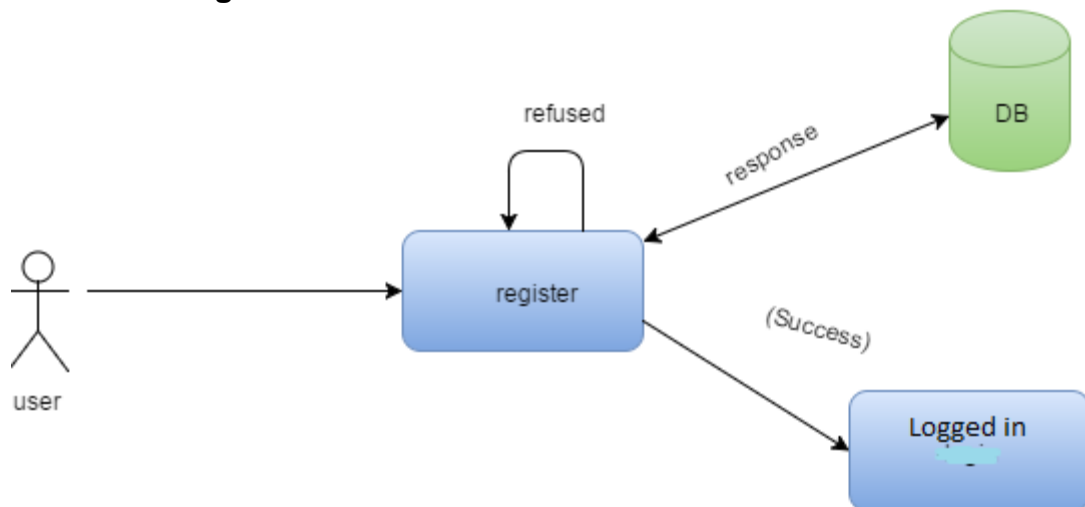
#### Scope

Register users

#### Description

Shows how user will be able to register onto the system

#### Use Case Diagram



#### Flow Description

#### Precondition

The system idle

#### Activation

When user clicks register button

**Main flow**

1. User clicks register
2. Sends username password and email (See A1)
3. User Logged in

**Alternate flow**

A1 : invalid credentials

1. User clicks register
2. The user username, email data is refused due to been used already
3. The User is asked to re-enter new credentials

**Termination**

User exits site

**Post condition**

The system goes into a wait state

**2.2.4 Requirement Leagues**

Leagues use case

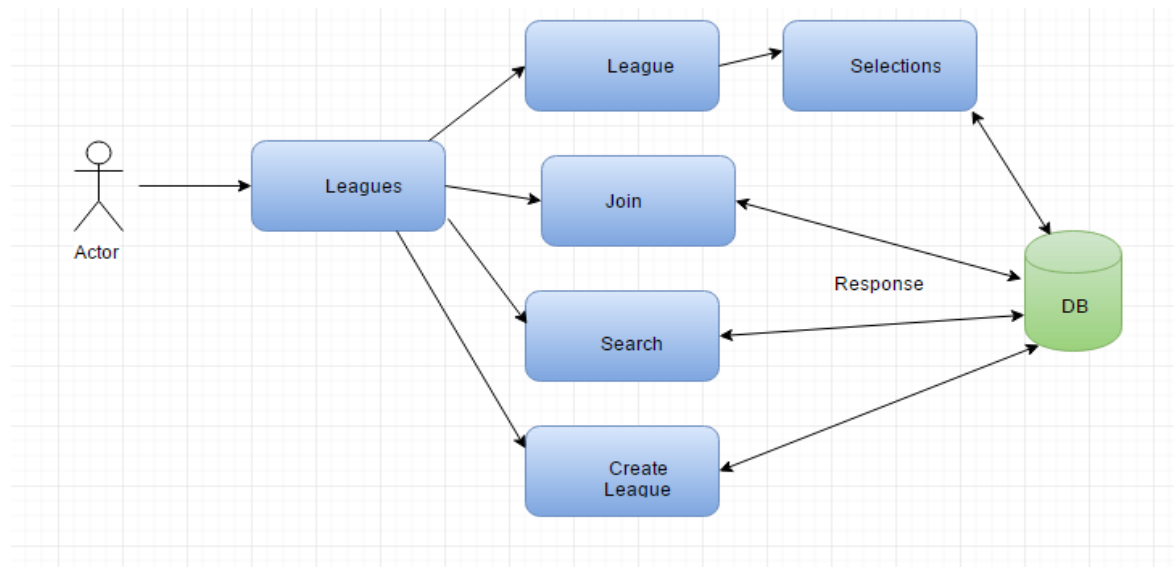
**2.2.4.1 Use Case****Scope**

League

**Description**

How the user will view leagues, join them , create them and search them

## Use Case Diagram



### Flow Description

#### Precondition

User logged in

#### Activation

When user clicks leagues button

#### Main flow

1. User click view table
2. User views League Table
3. User clicks create league (see A1)
4. User creates league sets private or public and league name
5. User clicks Join League (see A2,A3)
6. User search leagues

#### Alternate flow

A1 : invalid league name

1. User creates league
2. Refused creation
3. Asked to choose a different name

A2: League full

1. User clicks join League
2. User is refused as league is full

A3:Private league

1. Wait for admin to accept user to league
2. User Joins league

### Termination

User exits site

### Post condition

The system goes into a wait state

## 2.2.5 Requirement Profile

### 2.2.5.1 Use Case

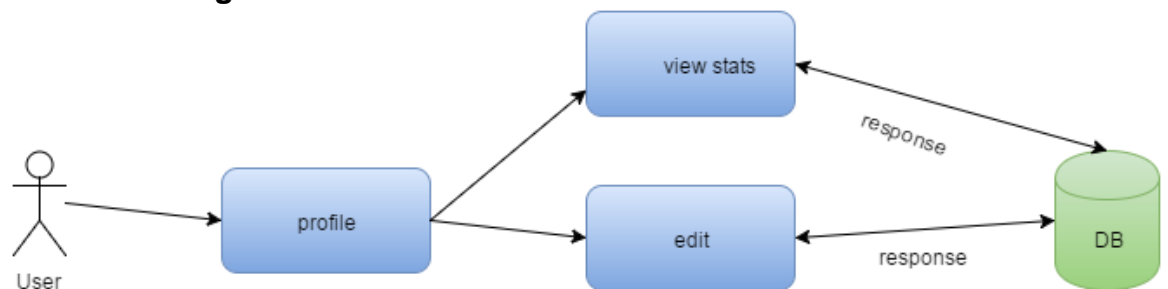
#### Scope

User profile

#### Description

Users profile page, edit profile and view stats

#### Use Case Diagram



#### Flow Description

#### Precondition

User logged in

#### Activation

When user clicks profile button

#### Main flow

1. User clicks profile
2. User clicks view stats
3. User edits profile (see A1)

### Alternate flow

A1 : invalid name change

1. User edits name
2. User edit team
3. Refused creation
4. Asked to choose a different name

### Termination

User exits site/logs out

### Post condition

The system goes into a wait state

#### 2.2.5.2 Use Case

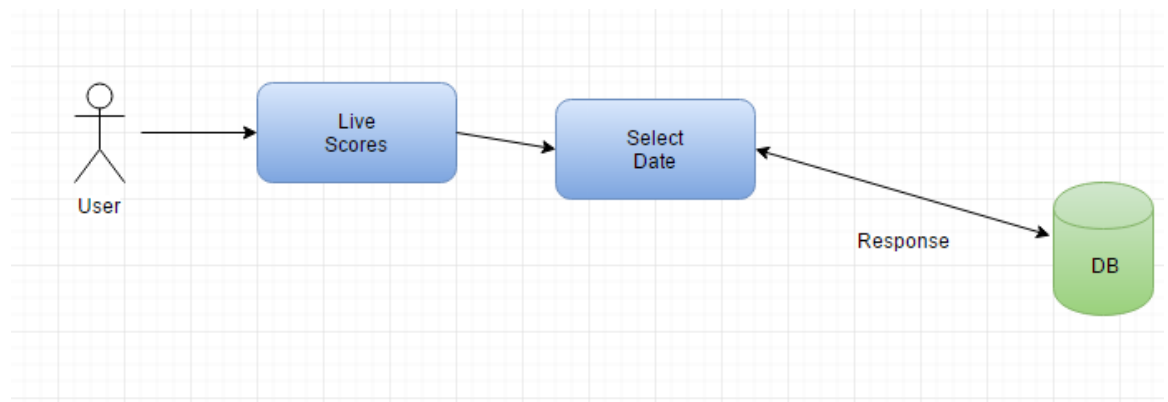
##### Scope

User Live Scores

##### Description

Users viewing live matches updates

##### Use Case Diagram



### **Flow Description**

#### **Precondition**

User logged in/logged out

#### **Activation**

When user clicks Live Scores link

#### **Main flow**

1. User clicks Live scores
2. User selects date of match
3. DB returns matches of that day

#### **Alternative flow**

1. User clicks live scores
2. User selects date of match
3. No matches available for that day

#### **Termination**

User exits site/logs out

#### **Post condition**

The system goes into a wait state

## **2.2.6 Requirement Selections**

### **2.2.6.1 Use Case**

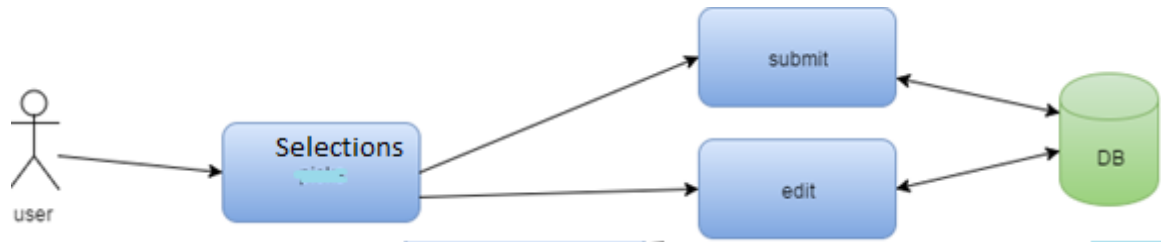
#### **Scope**

How the user selects their football selections

#### **Description**

How the user will select their picks and edit the picks up until match kick off

### Use Case Diagram



### Flow Description

#### Precondition

User logged in

#### Activation

When user selects picks and clicks submit

#### Main flow

1. User selects each match result
2. User submits the results
3. User submits new picks

#### Termination

User exits site/logs out

#### Post condition

The system goes into a wait state

## 2.2.7 Requirement Search

### 2.2.7.1 Use Case

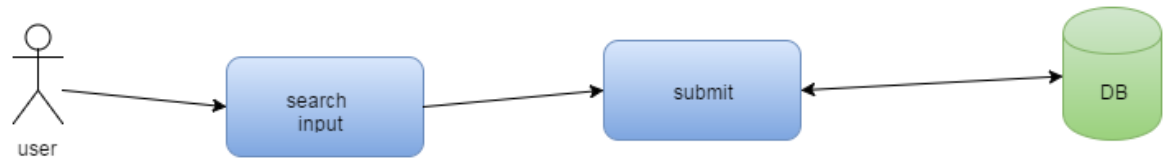
#### Scope

Search function for leagues and users

#### Description

How the user will search for other users and leagues by name

## Use Case Diagram



### Flow Description

#### Precondition

User logged in

#### Activation

When user clicks search button

#### Main flow

4. User types String into search bar
5. User clicks search button(see A1)
6. User can view results

#### Alternate flow

##### A1

1. User types String into search bar
2. User clicks search
3. No results returned

#### Termination

User exits site/logs out

#### Post condition

The system goes into a wait state

## 2.3 Non-Functional Requirements

### 2.3.1 Performance/Response time requirement



- The system will respond in real-time from 3- 5 seconds to update the view
- The system will write to the database within 3- 5 seconds

### **2.3.2 Availability requirement**

- User will be able to assess the app from a web browser
- Web browser will be optimised for tablets and different screen sizes
- User will be able to access the app via mobile application

### **2.3.3 Recover requirement**

- System will have a backup of database

### **2.3.4 Robustness requirement**

- System will recover from for server be down be restarting the server if the failure is detected

### **2.3.5 Security requirement**

- Web connection should use SSL connections, JSON web tokens for mobile

### **2.3.6 Reliability requirement**

- System will be running at all times
- User will have to log in again after crash
- 

### **2.3.7 Extendibility requirement**

- Can be used to extend the current sport by adding new games and newer sport leagues

### **2.3.8 Reusability requirement**

- Site will deployed to digital ocean, can be easily deployed to other cloud based server using Ubuntu virtual machine

### **2.3.9 Data requirements**

The data will be assessed via clients over http; both clients will send the data via JSON and will be return JSON data if requested.

The the database is be a NOSQL database called Mongo DB, the server will use a ODM called mongoose the will give the structure of the data in a schema, mongoose also offers other advantages.

### **2.3.10 User requirements**

The users require to have:

- **Android:** When using the mobile application the user must have access to an android phone with OS KitKat version 4.4 and above
- **Browser:** When accessing the web app, the user must have access to a web browser for example chrome, internet explorer fire fox and so on.
- **Internet Access:** Internet access is required to use all features of the application, data is access over http.

### **2.3.11 Environmental requirements**

- IDE: use a powerful IDE like intellij or a simple IDE such as web storm will be enough to develop and work with the web application, for using Ionic and building in Android, the Android IDE is needed to run and build android apps using ionic.
- Access to the internet for testing Api calls either mobile or web app
- Access to cloud hosting

### **2.3.12 Usability requirements**

- Easy to navigate the both applications, web and mobile
- Simple to understand where the user is on the application
- Users from 18 and onwards have no issue selecting games and leagues for

## ***2.4 Design and Architecture***

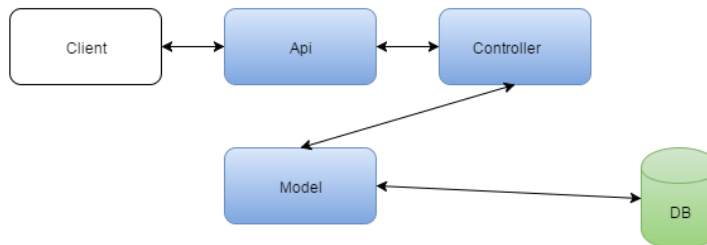
The goal of the web and mobile application was to create and easy way for users to send the selected picks either laptop, tablet or phone and for admins from their desktops.

The main web application was to build using the MVC architecture, it's a software architectural pattern that divides the web into Model View controller, and the Model is used for the DB, view to be serve to client and the controller to contain the logic.

The web app uses express which a restful frame work, the reason for using it was the app is designed to serve the web and mobile clients and it does this out of the box.

Also for the calculations of the player's scores, the league predictions will be run as their own instance and do not affect the web application. The reason for this is node , node is fast easy to code in JavaScript ,to reduce potential problems these will be separate and if the script breaks it will not brake the web app.

## Web MVC Architecture



### 2.5 Implementation

The use of Angular, Node js and mongo dB will be outline here and describing the technologies used to implement the Sport League's web and mobile application

To begin the process of building the application I installed node js globally, from there I create a new folder called sport leagues.

I used node js that comes NPM installed to download modules and frame works such as express and many other modules that help express js framework.

Once I download the necessary modules to get the web app running. I began working on the UI

#### 2.5.1 NPM

NPM is a package manger used for installing libraries for example Bower, mongoose.

I used NPM primarily for the server and used bower cli for the client libraries.

Jade a template engine was used for the UI.

## 2.5.2 UI

The web and mobile UI was created using HTML 5 and the bootstrap libraries, each view has templates which are included into one view, and this makes it easier for coding in specific areas of the UI.

Jade is used in the web app and is a cleaner looking html, below is an example of the league view and including other small templates.

```
div#mainPage
  .jumbotron
    .bg

    h3 Admin of {{league.leaguename}}
    h3 {{league.description}}
  .col-sm-6
    div (ng-include="'/partials/admin/adminForm'")
  .col-sm-6
    div (ng-include="'/partials/admin/leagueMembers'")
  .col-sm-6
    div (ng-include="'/partials/admin/requestToJoin'")
```

The Views in Angular js has the logic in the controllers, this is where the data is then served to the view.

Creating a new controller is simple, by creating a new JavaScript file and adding

```
angular.module('sportLeagueApp').controller('RegisterCtrl', function($scope,
AuthService, $ionicPopup, $state,$http)
```

After that you can use your factory singleton which I created call Auth service which contains logic for Authentication and saves the user to this factory and can be called in other controllers using authservice.

The web application mainly used the bootstrap library which makes the site mobile responsive and easy on the eye.

The mobile app uses Ionic and has built in views that make it look and feel like a native application.

## 2.5.3 Server

The server is built using the express framework, the clients connect to the server through its rest points, and this makes it easy to create new interfaces or new external clients to connect to the data.

The code below are an example of api endpoints which the clients connect to.

```
app.get('/api/user/:id', users.getUser);
app.get('/api/userInfo', users.getUsersInfo);
app.post('/api/users', users.createUser);
app.get('/api/selectteams', users.selectTeams)
```

## 2.5.4 Data and Mongoose (Mongodb ODM)

The data the football data is populated from an external api , the api supplies the data for the fixtures and the results of the matches.

This is an example of code been added to the data base.

fixtureSch.create is a mongoose method to create new document.

```
for(var i =0; i< fixtures.length;i++){

    fixtureSch.create({
        matchId:fixtures[i].id,
        week: fixtures[i].week,
        date: '',
        formatDate: fixtures[i].formatted_date,
        hTeam: fixtures[i].localteam_name,
        aTeam: fixtures[i].visitorteam_name,
        venue: fixtures[i].venue,
        hScore: fixtures[i].localteam_score,
        aScore: fixtures[i].visitorteam_score,
        matchStatus: fixtures[i].status,
    });

}
```

The fixtures object contains the data which i loop through the object to get each individual fixture information.

For updating the matches I filter through each using the method below the data from the mongo is contained in the obect called collect.

```
var match = collect.filter(function (match) {
    return match.matchId === fixtures[i].id ;
}).pop();
```

then I update the matches where the matches are not completed, 'FT' stands for full time.

```
//updating
if(match) {
```

```

if (match.matchId == fixtures[i].id && match.matchStatus != 'FT') {

    match.hScore = fixtures[i].localteam_score;
    match.aScore = fixtures[i].visitorteam_score;
    match.matchStatus = fixtures[i].status;
    match.save(function (err, updatedobject) {

        if (err) {
            console.log(err)
        }

    });
}

```

Once the data is insert or updated then i can show the user the fixtures and allow them to select the team the want

### 2.5.5 Users selections

The User sends his selections, the code below is from Angular js , the same code is implemented on mobile and web client and is used for adding new predictions and .

The code below loops to an array called \$scope.game, and if the user already made a prediction with the same id it removes the prediction in the array and then adds the new predictions to the array, if the id isn't in the array then addArray will be true and it will just add the prediction.

```

var addArray = true;

angular.forEach($scope.game, function(value, key) {

    if(value.id == id){
        var index = $scope.game.indexOf(value);
        $scope.game.splice(index, 1);
        $scope.game.push({
            id:id,
            pred:pred
        });
        addArray = false
    }else{

    }

});

if (addArray) {
    $scope.game.push({
        id:id,
        pred:pred
    });
}

```

Once the array is filled with the predictions the object is sent over via http post to the server using this code

```

$http.post('/api/selections/'+leagueId,$scope.game) .
then(function(response) {

    }
});

```

It posts over the object and league id is gotten on the server side.

```

League.findOne( { $and : [{ _id : req.params.id},
    {'users.userId': req.body.id}],
}).populate('league.users').exec(function(err, items) {

```

The server side checks to see if the user is in the league and if he is

Add the selections to the data base or If the predictions have been already been made.

Creating a user selection document

```

UserSelection.create({
    leagueId: req.params.id,
    userId: req.user.id,
    fixtureId: selections[j].id,
    pred: selections[j].pred,
    checked: false,
    correct: false
});

```

And here it's updating a changed selection

```

if (selections[j].id == collect.fixtureId && req.user.id == collect.userId) {
    collect.pred = selections[j].pred
    collect.save(function (err, updatedobject) {

        if (err) {
            console.log(err)
        }
    });
}

```

## 2.5.6 Scores and Calculations

The code below gets the users selections and compares it to fixtures results in the data base, it then returns the user selection changing the value correct to true or false depending on the outcome of the fixture. This only determines if the user is correct and doesn't add to the scores of the user,

```

if(collect !==null) {

    if (collect.hScore > collect.aScore && pred == collect.hTeam) {
        console.log(collect.hTeam, collect.aTeam, pred)

        collection.correct = true;

    } else if (collect.hScore < collect.aScore && pred == collect.aTeam) {
        console.log(collect.hTeam, collect.aTeam, pred)

        collection.correct = true;
    } else if (collect.hScore == collect.aScore && pred == 'Draw') {

        collection.correct = true;
    } else {

        collection.correct = false;
    }

    collection.checked = true;
}

```

The reason I wrote it like this was so I can add new functionality for game scores, such as bonus games or different types of competitions and calculating stats.

Example of calculating stats using count on the user selection based on been correct, user'id and the league and adds it back to the league.

```

UserSelection.count({$and:
[{'correct':true},{userId:user},{leagueId:id}]}).exec(function (err, collect) {

    collection.users[j].score = collect;
    collection.users[j].lastestScore = collect - points;
    collection.save(function (err, updatedobject) {

        if (err) {
            console.log(err)
        }

    });
});

```



## **2.6 Testing**

### **2.6.1 Usability testing**

For usability testing I asked users to use the site and while using the site they filled out a questionnaire, the questionnaire is based on Web Analysis and Measurement Inventory or (Wammi).

Wammi measures the usability of the web site while Wammi uses a 60 item questionnaire I wrote a small version but has the same principles,

Questions to find out if the site is attractive, do the users feel in charge while using, is the web site efficient in providing the information the user wants to see, can the user navigate the site with ease and if the user was to return could they use the site again.

The results of this testing can be found in the appendix

### **2.6.2 UNIT Testing**

I tested the server using Mocha js , the testing covered the rest points involved expecting a good response on each end point and testing for error handling that may have been missed.

Testing results are in the appendix

### **2.6.3 UAT Testing**

The UAT is effective because it's assumed the system is working and bug free, it's also used to identify the main working parts of the site, its most effective in the alpha and beta stages of the application and the decision will be made to push these features to the production application as long as they pass UAT, if bugs are found the bug will be left unless it's a major core part of the application and decide the seriousness of the bug.

All UAT testing was taken in Beta.

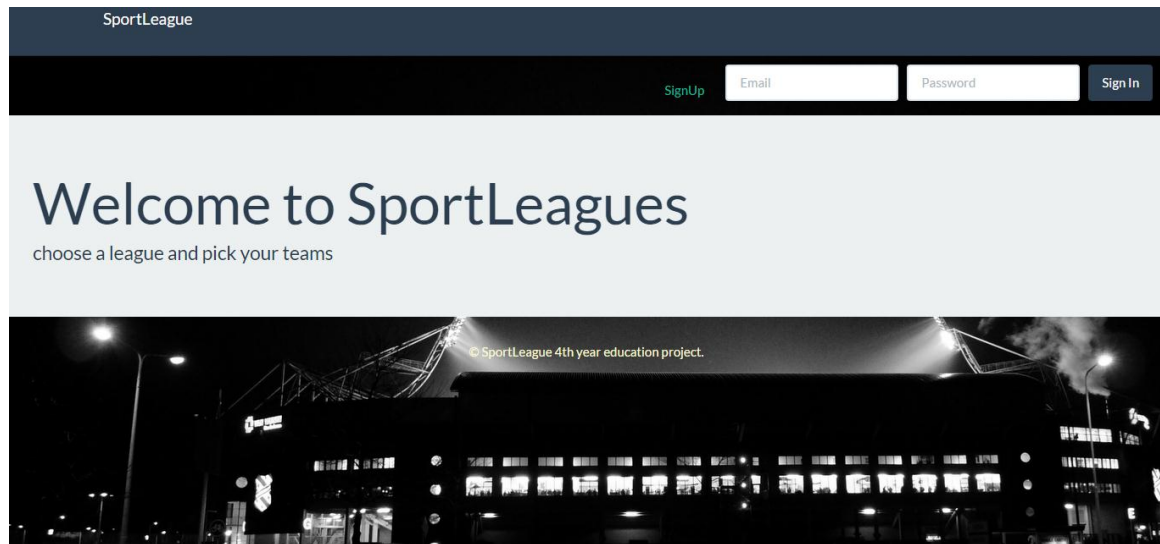
The users were using the application in the same environment as if they were customers.

The point of the UAT testing was to allow the users to complete a review of the site before new features are push to production.

The results of UAT testing in the Appendix

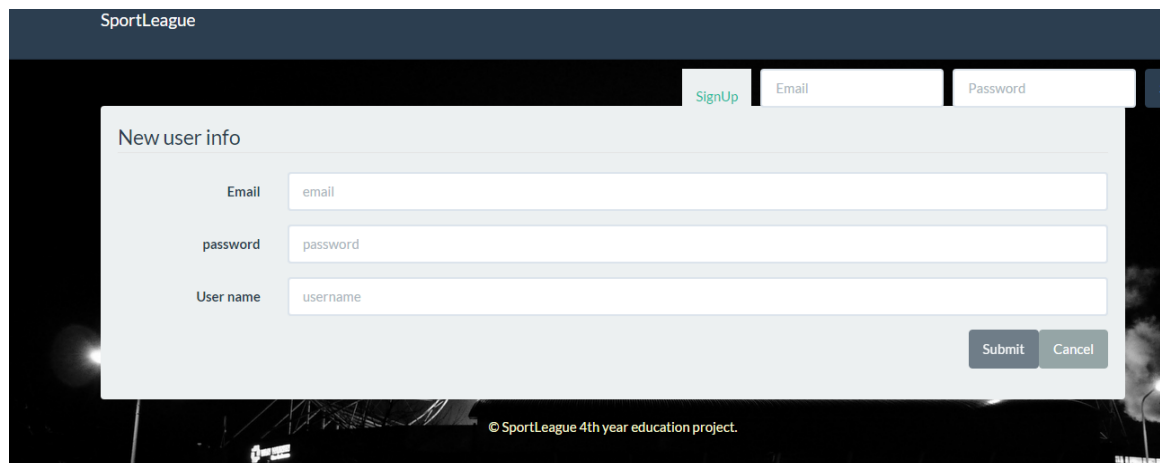
## 2.7 Graphical User Interface (GUI) Layout

### 2.7.1 Login



The screenshot shows the login page of the SportLeague application. At the top, there is a dark blue header with the text "SportLeague" on the left. On the right side of the header, there is a "SignUp" link in green, followed by input fields for "Email" and "Password", and a "Sign In" button. Below the header, there is a large light blue banner with the text "Welcome to SportLeagues" in a large, dark font, and "choose a league and pick your teams" in a smaller font below it. At the bottom of the page, there is a wide image of a stadium at night with lights. A small copyright notice "© SportLeague 4th year education project." is visible in the center of the image.

### 2.7.2 Register



The screenshot shows the register page of the SportLeague application. At the top, there is a dark blue header with the text "SportLeague" on the left. On the right side of the header, there is a "SignUp" link in green, followed by input fields for "Email" and "Password". Below the header, there is a large light blue banner with the text "Welcome to SportLeagues" in a large, dark font, and "choose a league and pick your teams" in a smaller font below it. At the bottom of the page, there is a wide image of a stadium at night with lights. A small copyright notice "© SportLeague 4th year education project." is visible in the center of the image.

New user info

Email	<input type="text" value="email"/>
password	<input type="text" value="password"/>
User name	<input type="text" value="username"/>

Submit Cancel

### 2.7.3 Leagues page

The screenshot shows a web interface for managing leagues. At the top, there is a light blue button labeled "Create a league". Below this is a dark blue header bar with the word "Leagues" in white. The main content area is a white table with three rows of league information. Each row includes a league name, its sport, and an "Invite only" button.

Leagues		
The Real League	Sport :rugby	<button>Invite only</button>
Tt	Sport :soccer	<button>Invite only</button>
Banter	Sport :soccer	<button>Invite only</button>

### 2.7.4 Create League

The screenshot shows a form titled "New League info" for creating a new league. It contains several input fields and dropdown menus. At the bottom right, there are "Create" and "Cancel" buttons.

New League info	
Name of League	<input type="text" value="league name"/>
League Description	<input type="text" value="Description"/>
Select Sport	<input type="text" value=""/>
Player Limit	<input type="text" value=""/>
Public or Private	<input type="text" value=""/>
Start Date	<input type="text" value="startdate"/>
<div><button>Create</button><button>Cancel</button></div>	

## 2.7.5 League

SportLeague

Go215

### The Real League

Barter

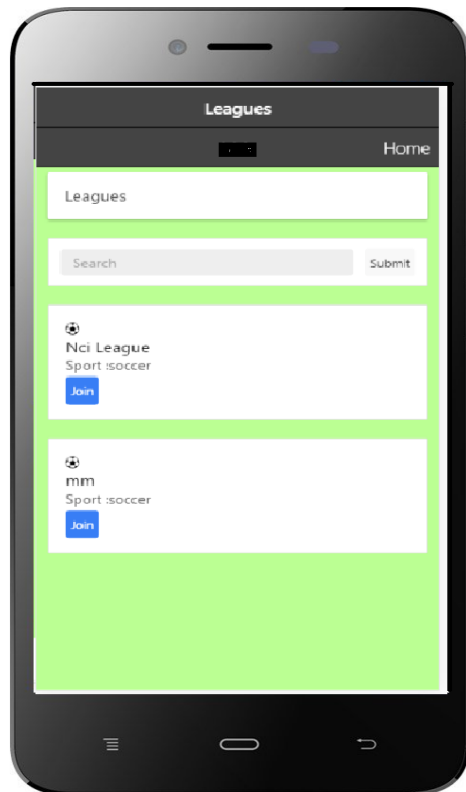
Picks

Win	Draw	Win
Manchester United result 3-0	Draw	Stoke City
Crystal Palace result 1-2	Draw	Bournemouth
West Brom result 1-1	Draw	Swansea
Leicester result 2-0	Draw	Liverpool

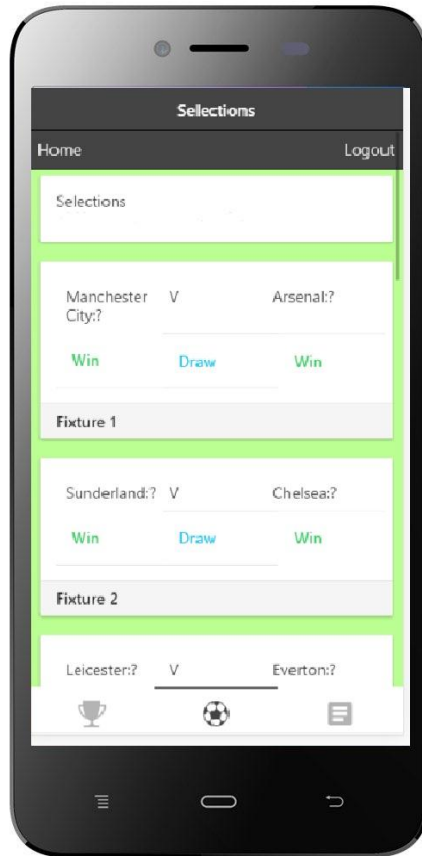
League Table

Pos	Name	Points	Form	Home	Away	Draw	GD
1	Leicester	50	WWWDW	14	2	8	18
2	Manchester City	47	WDWDW	14	5	5	23
3	Tottenham	45	WWWLD	12	3	9	25
4	Arsenal	45	DLDDW	13	5	6	15
5	Manchester United	40	WLWDW	11	6	7	10
6	West Ham	39	WDLWW	10	5	9	10
7	Southampton	34	DWWWL	9	8	7	8
8	Liverpool	34	LWLDL	9	8	7	-4
9	Stoke City	33	LLDWL	9	9	6	-4
10	Watford	32	WLLLL	9	9	5	1
11	Crystal Palace	31	LLLLL	9	11	4	-4
12	Everton	29	LDDDL	6	6	11	6
13	West Brom	29	DDLDDW	7	9	8	-8
14	Chelsea	28	WDDWD	7	9	7	-2

## 2.7.6 Mobile League



### 2.7.7 Mobile selections



## 2.8 Customer testing

To be carried out during Uat testing

## 2.9 Encountered problems and solutions

Example of mongoose ODM

The code below is example of finding if a user is part of a league already so when clicks the button to join the league it is not requested twice by mistake , if the code below returns a user within that league then it sends back a message to tell the client to hide the button for joining the league

```
League.findOne({_id: req.params.id}).exec(function(err, collection) {  
  //res.send(collection);  
  name = req.user.username.toLocaleLowerCase();
```

```

var person = collection.users.filter(function (person) {
  return person.username === req.user.username;
}).pop();

```

The code searches the league id sent of over via post request, it returns the object in the collection variable.

Using the session I can get the users name and set it lower case to match that in database just in case it may have changed.

Then from there mongoose has a filter function which loops through the users in the league and returns the object when the username match's the username in the object array.

If the user isn't part of the league then the object will be empty and using an if else statement the user will be able to join the league

```

User.findOne({_id: req.user._id}).exec(function(err, collection) {
  collection.leagues.push(req.params.id);
  collection.save(function(err, collection) {
  });

  })
collection.users.push({
  userId:req.user._id,

  username:name
});

```

The above code then finds the user and adds the object id to the user sub document users leagues in the users collection, this is done so it can be referenced

This is the current code as it stands and will be refactored for speed.

I will searching the users collection and checking their leagues array and comparing it to the id sent over via post request.

The reason for the change is mongoose allows you to reference other collections

```

var userSchema = mongoose.Schema({
  username: {type:String,
    required:'{PATH} is required',
    unique:true},
  email: String,
  salt: {type:String,
    required:'{PATH} is required',},
  hashed_pwd: {type:String,
    required:'{PATH} is required'},
  roles: [String],
  leagues:[{type: mongoose.Schema.Types.ObjectId, ref: 'League' }]
});

```

The code above is the user schema, and as you can see in the leagues array it contains a ref to the leagues collections I only discovered you can use this to call a user collection and populate the collection with data from the leagues collection without having to makes quires to other collection yourself.

According to mongoDb each collection has a limit to 16mb so only using id's instead of all the league data saves space.

## Conclusions

### Achievements

The project has achieved most of the goals I set out.

The client was happy with the progress of the application and will be using this success to further develop the application over the coming months.

The UAT testing and other testing has found the site to be a success to the market target.

I am extremely happy with the outcome of the project and I pushed myself to in terms of code and building an application of this size.

This project alone has increased my confidence in coding ability and adding new features that I would have believed to be too hard to undertake.



### **3 Further development or research**

This project can develop into using other sports and leagues as the architecture allows for different sports to be slotted in using the same of similar games rules.

There also can be a business model of users paying for extra features for better entertainment as well as taking a percentage of gambling winnings from users.

## 4 References

### 4.1 References

- "Angularjs — Superheroic Javascript MVW Framework". *Angularjs.org*. N.p., 2016. Web. 11 Aug. 2015.
- "Building Angular And Node.Js Apps With The MEAN Stack | Pluralsight". *App.pluralsight.com*. N.p., 2016. Web. 11 Sept. 2015.
- "Building Mobile Apps With The Ionic Framework And Angularjs | Pluralsight". *App.pluralsight.com*. N.p., 2016. Web. 11 Feb. 2016.
- "Continuous Integration And Deployment For Angularjs And Node.Js | Pluralsight". *App.pluralsight.com*. N.p., 2016. Web. 11 Feb. 2016.
- "CSS Components - Ionic Framework". *Ionicframework.com*. N.p., 2016. Web. 11 Mar. 2016.
- "How To Build And Test Your Rest API With Node.Js, Express And Mocha". *thewayofcode*. N.p., 2013. Web. 11 Dec. 2015.
- "Mongoose ODM V4.4.16". *Mongoosejs.com*. N.p., 2016. Web. 11 Sept. 2015.
- "Mongoose Query Population V4.4.16". *Mongoosejs.com*. N.p., 2016. Web. 11 Feb. 2016.
- "Mongoose Subdocuments V4.4.16". *Mongoosejs.com*. N.p., 2016. Web. 11 Mar. 2016.
- Node.js, How. "How Do I Get Started With Node.Js". *Stackoverflow.com*. N.p., 2015. Web. 11 Oct. 2015.
- "Football API - Affordable Football Live Feed For Developers". *Football API*. N.p., 2014. Web. 11 Sept. 2015.

Ian Costello

Technical report midpoint . Print

## **5 Appendix**

### **5.1 *Project Proposal***

#### **Objectives**

- Friendly UI
- Instant updates of results and User results
- Social gambling entertainment
- Backend to work with Any frontend and client based applications using restful web services

The user will be able to:

- Join leagues
- View player leagues
- User v User (player V player)
- Invite players
- Create leagues
- View premierships football results fixtures and league standing's
- Chat to other users
- Social logins and sharing
- View other player stats and results
- View own stats and results

#### **Background**

The plan is to build a web and mobile application using restful services, this application will be design so it can be used with other services and possible developed into a working product for the gambling industry. The online gambling industry is growing business and I believe automating games people play with other like last man standing and so on, the user can have a better experience. Also adding the social nature with games is a must which is rarely seen in the likes of paddy power or Ladbrokes.

The project is s called sport league, a very generic name for a project but I don't want to infringe on other companies' names.

The idea is to do with premiership football, each week there is 10 football matches, people can choose the outcome of the match by either choosing a home win Draw or away win. The image below is an example.

West Brom	Draw	Sunderland	Oct 5
Spurs	Draw	Saints	Oct 5
Manchester Utd	Draw	Liverpool	Oct 5

The image is mine, it's a screen shot from my site in development.

The users will be allocated points for each prediction and they shall be presented the results in a very user friendly UI in web and Mobile, the results will be shown in the table from first to last.

The user will be enter into a league table with other users, and after a set period of weeks a winner will determined by coming 1<sup>st</sup> and top of the league.

Also I wished to implement is a User v User, in which a player can face any other user that week to determined who is the king of picking winners.

I also wish to build social functionality into the application most likely with chat for each league or maybe over selected matches while in play.

## Technical Approach

I wish to make a web based app as well as mobile, the backend will be using restful web services so I can use the same data and backend while been able to have different frontend and clients.

The backend will be hosted in the cloud and requests can be made from there, the frontend will be mobile friendly web app and also if it's possible build a mobile app as well.

The backend will consist of Node.js using the express framework, Node while new, has been proven effective and used by google Microsoft, Linkeden etc,

The reason I choose node was for its real time capabilities for pushing and updating data and it's easy to configure and speed at which you can get your project up and running compared to for example Spring.

For the front end I have chosen to Angular js, this is a popular framework and uses data binding to update the client,

Db will most likely be MongoDB as it's fast and easy to create collections without designing a schema, but in this case I will be designing a schema except it's easier to update the schema as the project unfolds using MongoDB

The tools il be using are Web Storm for both Node and Angular and possibility for another framework for mobile.

All data will connect by sending JSON to the clients.

The football results will be collected from an 3<sup>rd</sup> party Api web services, which I will store the results in my Db and then use the data as necessary , by comparing user results with the results of the football matches.

## **Resources**

As I know of I will be using Plurasight for tutorials.

Most likely ll check and use the documentation of Node.js, Angular.js and any other modules from NPM or language's I will use.

I will post and record and other tutorials videos and books to

## **Project Plan**

See appendix

## **Technical details**

Javascript: Node.js ,Angular.js, Express, Mongoddb, Karma.js, Mocha.js.

Cloud hosting: Digital Ocean

OS: Linux

Git: GitHub

Moblie : Meteor.js or Ionic Framework

## **Evaluation**

I will be using testing tools such as karma for the client view and mocha for testing the express application, other testing methods like for the UI and UX I will be asking different types of users from different age groups to test the functionality and the ease of use, I will document this and show it in my report as well as updating the application with the new solutions.

Plan	Actual	%Complete	Actual (beyond plan)	%Complete (beyond plan)

- 47 -

### 5.3 Monthly Journals

## Reflective Journal

---

Student name: Ian Costello

Student number: 12112348

Programme (BSc in Computing mobile and networking):

Month: September

Began to test sample apps using the football Api from football-api.com,

The offer 1000 calls per day and offer the premier league fixtures and results for free, the apps I was testing it was Node js and Express framework and using with Angular js for the frontend.

I got the APi to work and to store into the database, the data base I was using was Mongo Db, because I got I connected I was designing sample schemas for the real project

I used tutorials on Pluarsight to build the sample apps and github to store the sample apps and host it to Heroku.

Discover that GitHub have a student discount pack and offer 100 dollar gift voucher for using Digital Ocean cloud hosting, so I've decided to use that as my hosting service for my application.

Also in two minds about doing a mobile application web or Both, unsure of the complexity of such a project, so I've decided to build the backend in mind of using both so I need to I can build a mobile Application.

My Achievements

Begin coding examples of sample apps, finding online tutorials for the mean stack( JavaScript frameworks and other libraries ), connected to the database on my local machine (mongo Db)

Contributions I've made is setting up the server

Deciding the web technologies

Designing the schema of my database

Worked with the API which will give the results of the football matches and fixtures in JSON

Hosted project to GitHub

Design parts of the UI



### My Reflection

Great month over all, I feel happy with my start as I know exactly what I plan to do and for the most part I know how to execute the project, I've given myself plenty of time work on the project which gives me the ability to overcome unforeseen obstacles that exist in building software.

### Intended Changes

It will be deploying my project to the cloud in the next month and hopefully will have an login system for users as well as registration for user as well as storing the users picks and displaying the information.

## Reflective Journal

---

Student name: Ian Costello

Programme: BSc in Computing

Month: October

I have built my login system for both the front and backend, I've set the user modal in the Db so now a user can create an account and log in. I've created the user profile page in angular and it receives data from the DB.

User profile page designed, also user picks paged design with premier league table working from football Api.

I haven't had much time to deploy to the cloud as I have realised the database models are

unfinished and wanted to focus more on the design and architecture of the application.

I used Pluralsight heavily that month and have also bought two Udemy Node.js tutorials, they go in to detail about how to get the very best from node.js.

I've spoken to Francis Sheridan, and Francis insists on making an mobile application and saying I will have enough time to build the mobile application alongside the web app.

She also told me if I get into trouble I could make a slim down mobile app compared to the web app, as long as I have decent argument for it.

I researched other similar apps fantasy football and going to base similar set to my application with my niche features that make sit stand out.

This month has been quite business with other modules in the course so I am planning to focus a lot more time on my project and set new goals.

### **My Achievements**

The core of the application has begun with creating a user and login system built, researched mobile technologies such as Android, meteor js, Xamarin.

- Built login system
- Built registration system
- More mock ups of web site pages
- Used football Api to show alongside user picks
- Login and registration web services built

### **My Reflection**

Didn't get to deploy the application to the cloud as previously thought, setbacks from other college modules and also not a finished requirement spec made decide to delay it until the requirement spec was finished.

I have a client who I am building the application for and I was waiting on certain aspects the meeting to update my requirement spec.

My client wishes to more control over the point system so each game rules can be changed with each league table. This the way he has set it up, but there may be new conditions for group v groups within the leagues. Which I have set out in the requirement spec.

### **Intended Changes**

Changes to UI design as the req spec has changed , so resigning certain backend functions and changes to data modals

### **Supervisor Meetings**

Date of Meeting: Tue 10/20/2015 2:18 PM

Items discussed: My idea, complexity of the idea, and the use of a mobile framework.

Action Items: Research other similar applications; Research mobile tech

# Reflective Journal

---

Student name: Ian Costello

Programme: BSc in Computing

Month: November

I have created the create league page which allows the user s to join the league, the functionality is not done for joining the league but it's on next step for building the application.

I have discovered other external Api's that give gambling prices for each match and I am considering adding it in once the client is happy.

I've spoken to Francis Sheridan, again about documentation and she seems happy about documentation and I have uploaded my project analysis as work in progress.

This month has been quite business with other modules in the course so I am planning to focus a lot more time on my project and set new goals.

## **My Achievements**

- Built league pages for the users to join

## **My Reflection**

I have been extremely business with other modules I feel I have not meeting my goals , I will make sure I have a complete proof of concept and work my hardest over the Christmas period.

## **Intended Changes**

As of yet none

## **Supervisor Meetings**

Date of Meeting: Tue 12/04/2015 2:18 PM

# Reflective Journal

---

Student name: Ian Costello

Programme: BSc in Computing

Month: December

Building chat for league and experimenting with socket.io

Also researched and tired out using socket.io

Laptop has broken in the final week. Not good news.

#### **My Achievements**

- Building example chat system

#### **My Reflection**

Massive set back with the loss of my laptop, trying get data back and work I've saved.

Not a good month

#### **Intended Changes**

As of yet none

#### **Supervisor Meetings**

Date of Meeting:

Items discussed: Documentation.

# Reflective Journal

---

Student name: Ian Costello

Programme BSHC in Computing 4th

Month: January

I discovered problems with how I was searching my data using mongo db , this is all a learning curve for using a nosql database, I came across a problem that needed joins and can be solved easily with SQL.

The solution was that the ODM mongoose has built in referencing, once I researched it and came figured how to implement it I refactor a lot of my code reducing loops and a easier to read code base.

I also figured out how to install ionic on my machine which I had trouble over the Christmas period.

Now I have it installed I tested it be creating a simple app with a list for showing data, the ease of use now gives me confidence that I can create a mobile application in such a short time as well as a web application.

Third thing I complete this month was hosting my project online on the cloud via Digital Ocean, the virtual machine uses linux and great learning curve using the commands and working environment.

Hosting application allows me now to build the mobile application and test the web server restful end points.

Also the server will be used for User acceptance testing, so users can give me feedback after trying out use cases on the site.

## My Achievements

Database research, cleaner code faster queries.

Fixing mobile development issues so work can begin on mobile app.

Hosting the application so it can go in beta and receive user testing.

## My Reflection

Over all great month, a bounce back from last month were my laptop broke and I lost some work.

### Intended Changes

Major parts coming together now so I must focus on the game logic for calculating scores and develop the mobile app now since it can be tested.

### Supervisor Meetings

Date of Meeting: 29/1/2016

Items discussed: Showing of prototype, talking of what stage the app is in

Action Items: Develop a mobile application

## Reflective Journal

---

Student name: Ian Costello

Programme BSHC in Computing 4th

Month: February

During my presentation I was told to get cracking on the mobile development side of my project, I've started the login and registration and which is almost completed the next step for this is to send over the user selections, using ionic is handy due to the familiarity of using angular js.

I ve also worked stats and calculations for the game logic, while the over logic works by using one process to determined is the user selection was correct but changing a Boolean to true in Db and then I have a separate process the counts the scores .

This allows me to add more ways to get stats. These process are separate from the web server process and will run by them self's.

Also Im using Cron to get regular fixtures updates and live updates for footballs matches.

I've spoken to my client and he has told me while he would like messaging within the application to other users he said most likely people would use other apps for chatting and to focus on other aspects of the application. I ve agreed with him and have made it a low priority.

As I can see now once I can get the mobile application complete before march or early April I can finish the rest of the web app.

## My Achievements

Login and registration for mobile application

First set of stats created and game logic implemented

## My Reflection

Over all great month can really see the project coming together now

## Intended Changes

Work on mobile application up until late march early April and then from there finish the web app

## Supervisor Meetings

Date of Meeting: 12/2/2016

Items discussed: Project presentation and where my application stage is at

Action Items: finish mobile application

# Reflective Journal

---

Student name: Ian Costello

Programme: BSHC in Computing 4th

Month: March

The chat functionality has been removed and it's not a low priority.

Work on the mobile has been great, the user can get leagues check their profile and send over their football predictions, there has been issues with cross origin requests due to testing ionic with the browser but was easily overcome, also Data and cache issues. The issue caused angular not save and show the new user logged in if another user logged out. Ionic does this for performance but it's not necessary for my app as it isn't image heavy and data heavy.

Other issues I the External api to receive football stats are in the process in changing to API version 2. There changing the data layout which I have to accommodate for this but I will wait until they have completed the change.

I have been clean up backend code for the Admin services and stats, I have the UI for this but implementing the features for getting stats and allowing users join and be refused if the league is private.

### My Achievements

User gets profile, leagues and sends football predictions for mobile application  
Beginning work for Admin areas for the leagues and stats.

### My Reflection

Over all great month can really see the project coming together now

### Intended Changes

Complete outstanding features such as admin

### Supervisor Meetings

Date of Meeting: 12/2/2016

Items discussed: Project presentation and where my application stage is at

Action Items: finish mobile application

## Reflective Journal

---

Student name: Ian Costello

Programme: BSc in Computing

Month: April

Cleaning up website, completing features for the admin section, this was for changing the name of the league changing if the league is private or public and changing the description of the league,

Also the admin layout of the page, which includes existing members and users who request to join a league.

Also clearing up the rest points for mobile and testing rest points making sure the token requests work with each request.



Found a bug that that affect selections which in certain cases did not return a response to the user.

Bug was fixed by using `res.end()` for Express js.

Also completing the live score requirement which allows using to select matches in between 10 days from today, allowing them to see last week's results, live scores today and see future fixture's .

The dates come from the server to the client and the user then picks a date and is returned the fixtures on that day , these can be finished matches or future fixtures.

Implementing testing using Mocha js.

Getting results back from UAT testing and Usability testing.

This consisted giving the use test cases for UAT and Usability give the users a series of questions of the web site and page and they have to describe the their results.

### **My Achievements**

Implementing final features the requirements of admin

And live score feature where user can select football matches by the day to see results or future fixtures.

### **My Reflection**

Great over final month , just adding finishing touches to the technical document and making the site UI look better, getting good reviews for users and ideas for future improvements or even new features.

Just ready now to do my poster and upload my project.

### **Intended Changes**

Changes to UI design adding images

### **Supervisor Meetings**

Date of Meeting: Tue 27/04/2016 2:18 PM

## 5.4 Unit testing with mocha js

The example code is testing the rest point /api/leagues, the function and the DB.

What im looking to test is the values that are sent over are sufficient to work and is looking for the response 200 for the test to pass.

The code failed and rewrite has been taking place and tested again.

Example of response

```
0  _ _ |  / \ _ \
1  ^ | _ _ ( - . - )
   "" ""
0  _ _
   _ _
   _ _ , ----- ,
   _ _ |  / \ _ \
open ~ | _ ( x . x )
      "" ""
```

0 passing (103ms)

1 failing

1) get selections data should return user data:

```
describe('get selections data', function() {
  var req = {};
  var res = {};
  req.user.id='34'
  it('should return user data',function(done) {

    var selections = {
      id: '1234',
      pred: 'draw',
    }
    agent.post('/api/leagues')
      .send(req,res)
      .expect(200)
      .end(function(err,results) {

        results.body.read.should.equal(true)
        // results.body.should.have.property('_id');
      })
  })
})
```

```
done();
});
```

The below is another example , the code passed the test first time, it was to test the end point for teams been selected or edited on registration and profile.

```
describe('get teams for user to select', function() {
  var req = {};
  var res = {};
  req.user.id='34'
  it('should return user data', function(done) {

    agent.post('/api/selectteams')
      .send(req, res)
      .expect(200)
      .end(function(err, results) {

        results.body.read.should.equal(true)

      })
    done();
  });
});
```

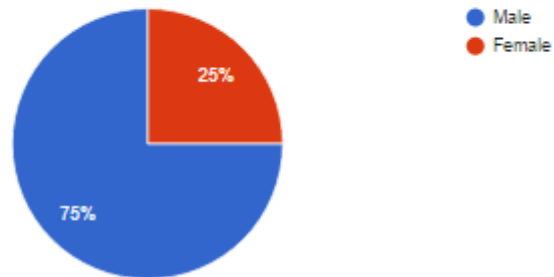
## 5.5 UAT testing results

Number	Acceptance Requirement	Critical yes	Critical No	Accept	Reject	Comments
1	User can log in	True		True		Login is essential
2	User can join league	True		True		No response when joins, but works.
3	User can create League	True		True		
4	User Can send selections	True		True		

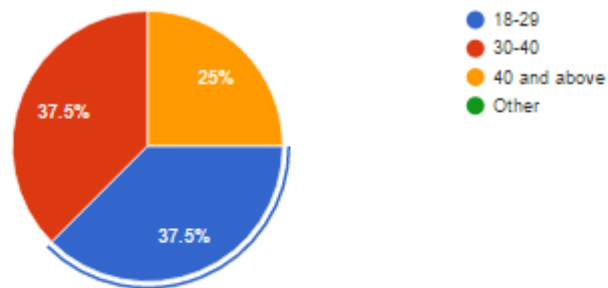
5	User can select matches by date		false	True		
6	User can search leagues	True		True		
7	User can register	True		True		
8	Admin can edit leagues	true		True		

## 5.6 Usability testing results

Gender (8 responses)



What is your age ? (8 responses)



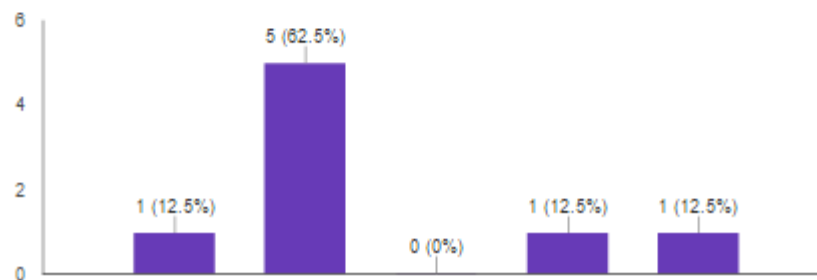
Whats your reason for visiting this web site? (8 responses)

Interest in soccer
The love of football
Sports
Gambling
banter
betting
football fan
sports fan

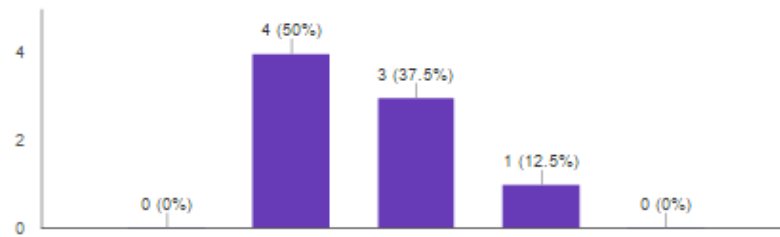
(0 responses)

No responses yet for this question.

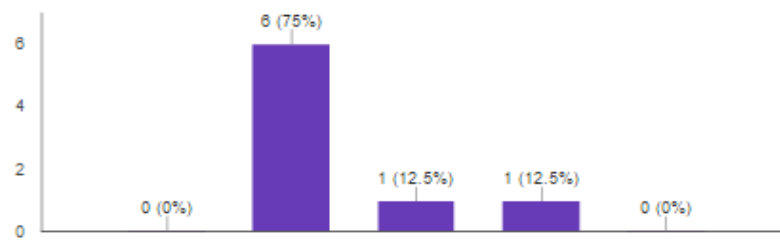
The web site has sports that interest me (8 responses)



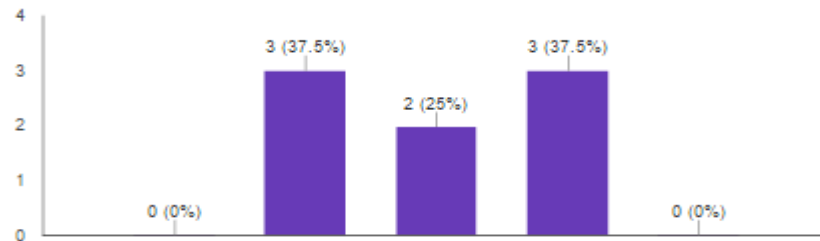
This web site needs more explanations (8 responses)



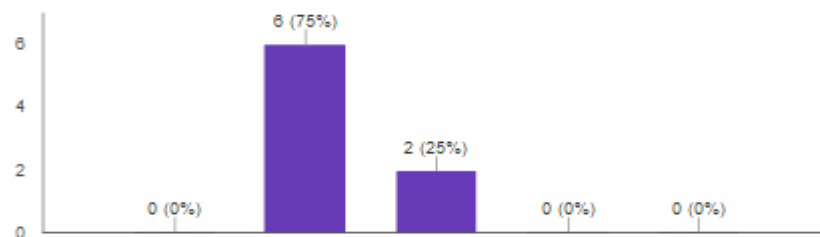
This web site helps me find what i'am looking for (8 responses)



its difficult to navigate around this web site (8 responses)



This web site seems logical to me (8 responses)



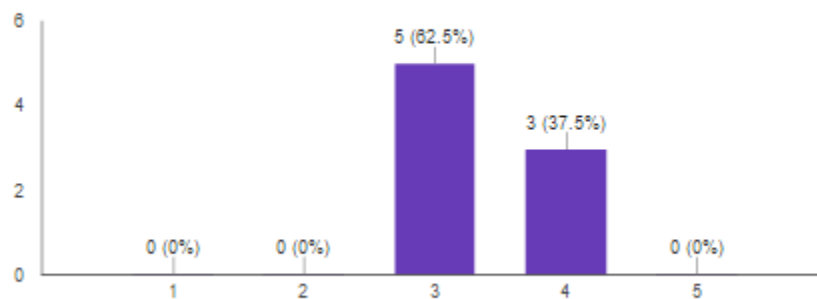
whats the best feature on this web site? (8 responses)

The league tables
The social aspect
fan involvement
fan factor
The leagues
the results
The layout
The scores

What can be improved of this web site? (8 responses)

UI
More games
More football competitions
cash bets
better UI
payments system
live scores for users
More stats

Can you rate this site (8 responses)



(0 responses)