

**ASSIGNMENT COVER SHEET**

**Student Name Michael James, Martin Zuber, David Kelly**

**ID Number: B00019330**

**Course: Computer Science**

**Year: 1 of 1**

**Lecturer: Luke Raeside**

**Title of Assignment: Thesis Document**

**Due Date: 1/5/15**

**Date Submitted: 1/5/15**

The material contained in this assignment is the author’s original work, except where work quoted is duly acknowledged in the text. No aspect of this assignment has been previously submitted for assessment in any other unit or course.

**Contents**

[**Project Description** 3](#_Toc415077738)

[**What We Will Look at in this document** 3](#_Toc415077739)

[**Development Methodology** 4](#_Toc415077740)

[**Project Plan** 6](#_Toc415077741)

[**Use Cases** 7](#_Toc415077742)

[User checks current score 7](#_Toc415077743)

[User checks fixture 8](#_Toc415077744)

[User uses social media aspect 9](#_Toc415077745)

[User creates team 10](#_Toc415077746)

[User makes transfer 11](#_Toc415077747)

[**Sequence Diagrams for Use Cases** 12](#_Toc415077748)

[User checks score 12](#_Toc415077749)

[User checks fixtures 12](#_Toc415077750)

[User posts message 13](#_Toc415077751)

[User chooses team 13](#_Toc415077752)

[User makes a transfer 14](#_Toc415077753)

[**Wire framing** 15](#_Toc415077754)

[Login Page 15](#_Toc415077755)

[Registration Page 16](#_Toc415077756)

[Team statistics page 17](#_Toc415077757)

[Player transfer page 18](#_Toc415077758)

[Standings Page 19](#_Toc415077759)

[Social Media area 20](#_Toc415077760)

[**Work Flow Plan** 21](#_Toc415077761)

[**DESIGN and TECHNOLOGY details** 22](#_Toc415077762)

[**References** 25](#_Toc415077763)

## **Project Description**

The project we are creating and that has been proposed in previous documentation is a fantasy hurling game. There will be a social media aspect to the application too. The game allows you create an account, then choose a squad of players from a list. You have a maximum budget and must stay under this. Each player is awarded a score each match based on real life GAA championship games. Each user’s team is then updated at the end of each game round. The main technologies used will be Java for the backend, html and JavaScript with some CSS for styling for the front end, and of course the database, which will be SQL.

There should be no real compatibility issues within the project, however a separate mobile version of the application could, at a future time be developed. There will be no special requirements for usage of the product. It will be web based as in it will be accessed entirely via the normal HTTP request in the browser, no special software needs to be installed, and no special hardware is needed for the application on the user’s side.

On the server side, we will host the application on Microsoft’s azure servers.

## **What We Will Look at in this document**

In this document we will provide some analysis and conceptual design of the product using UML diagrams and modelling techniques. We will look at the following:

* Comprehensive Use Case Diagrams
* Development Methodology
* Sequence Diagrams
* Class Diagram of the final product
* Activity diagrams
* User Interface Design
* Database table and schema design and description

## **Development Methodology**

In this section we are going to discuss the methodology of software development life circle (SDLC) we chose to use to develop this project. After initial research we decided for Prototyping SDLC.

**Prototyping SDLC, what is it basically?**

In this approach the development team implements a ‘sample’ which have only very limited functionality of proposed project and show it to the customer. Customer provides the suggestions of improvements and finally the development team implements these. This circle then repeated itself numerous times until the project is fully functional, tested and accepted by customer. This approach differs from other methodologies by avoiding doing the ‘big design in advance’ followed by implementation, testing and deployment phase. The project is rather developed by mutating the prototype with numerous design, implementation testing and deployment phases until the final product is build ***[1]***.

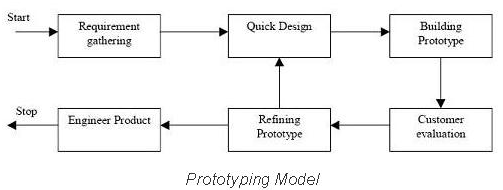


Figure 1 Prototyping SDLC work flow

**Why did we decided for this approach?**

We decided for prototyping approach for various reasons

* We wanted to avoid ‘big design in advance’. To design web application in advance it requires experience. Only experienced developers who already worked on projects with similar functionality and scope are able to do that. If ‘an amateur’ tries to design the application he will realize during the implementation phase that his design has flaws and redesign is required. This will lead to lose of work hours and any codebase the developer had already implemented. Basically you need an architect to design the house. We amateurs. We never build web application of this scope. With prototyping approach we be able to redesign the sample as many times as needed.
* We will learn with each prototype iteration. In our case we first build a simple but working sample build from prototypes. Data store prototype (relational database layer), data model prototype (objects to data mapping layer), controller prototype (business logic layer), and the view prototype (front end client layer). We make sure it all works together. Then we pick one proposed functionality and implement it whole way down through all the layers. And we learn from it. Implementation of next functionality will be easier and we get more productive over time. Hopefully after various iterations we will be able to call ourselves ‘the web developers’.
* Time restrictions. According to our project plan the first working prototype delivery is due to 14.February 2015. But we have been notified (by email 14.November 2014) that we must deliver the ‘working prototype’ due to 19. December 2014. According to our plan this is a two months ahead. Being still in research phase of project plan we assessed that only feasible approach under these circumstances is prototyping SLDC.

## **Project Plan**

*In this section we are going to discuss how we planned to develop this project.*

**What are we developing?**

Fantasy hurling project is basically fantasy sport game. It’s rich web application not a static web page. As a player manages his fantasy hurling team a content dynamically changes. At the same time it must have a persistence capabilities. Basically a changes player made must be stored somehow and not get lost when player finishes to play. Data must outlive the application runtime.

*Basically we will need to have three main layers.*

* Front-End (code running in browser delivered by initial http request from web server)
* Back-End (code running on server listening and answering to http requests)
* Persistence Storage (relational database server listening and answering CRUD queries)

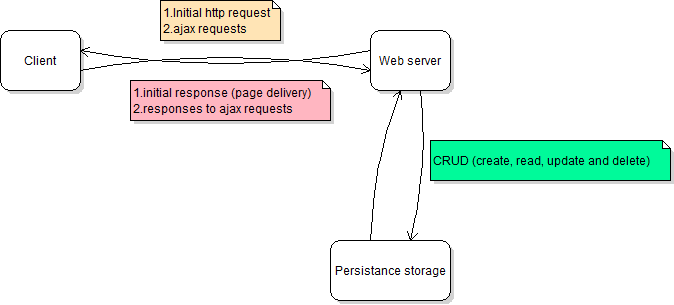


Figure 2 Web Application general design

## **Use Cases**

### User checks current score

Use case specification:

1: User logs in

1a: User logs off

1b: User not registered and is prompted to register

2: User checks their current score



### User checks fixture

Use case specification:

1: User logs in

1a: User logs off

1b: User not registered and is prompted to register

2: User checks fixtures coming up



### User uses social media aspect

Use case specification:

1: User logs in

1a: User logs off

1b: User not registered and is prompted to register

2: User sends message

2a: User posts in the forum

2b: User sends an instant message



### User creates team

Use case specification:

1: User logs in

1a: User logs off

1b: User not registered and is prompted to register

2: User makes team

3: User chooses squad

4: User chooses team name



### User makes transfer

Use case specification:

1: User logs in

1a: User logs off

1b: User not registered and is prompted to register

2: User makes transfer

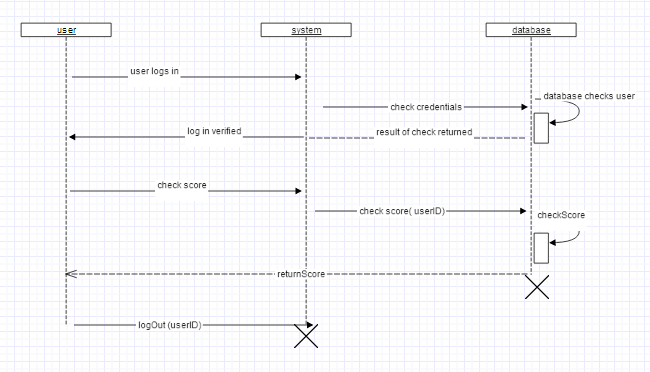
2a: User has not enough funds to make transfer

2b: User selects and invalid team choice

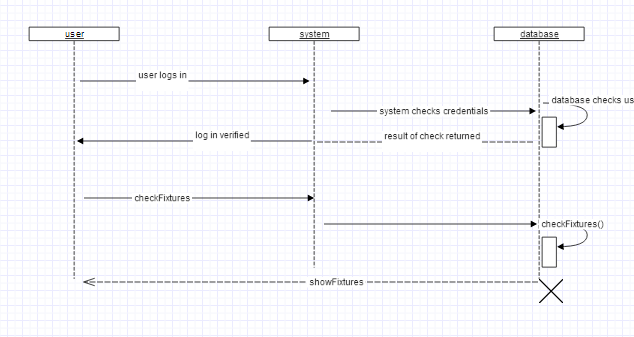


## **Sequence Diagrams for Use Cases**

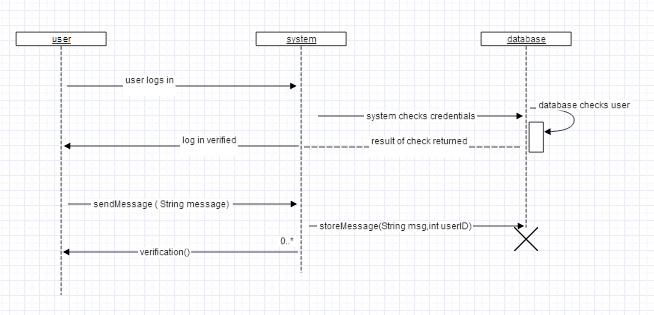
### User checks score



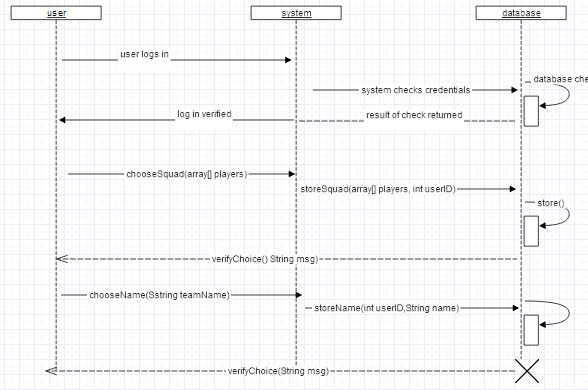
### User checks fixtures



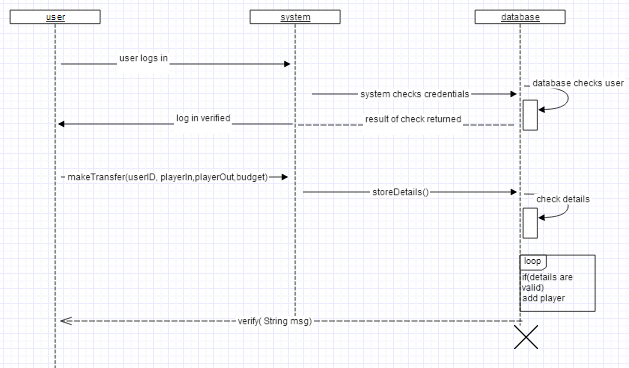
### User posts message



### User chooses team

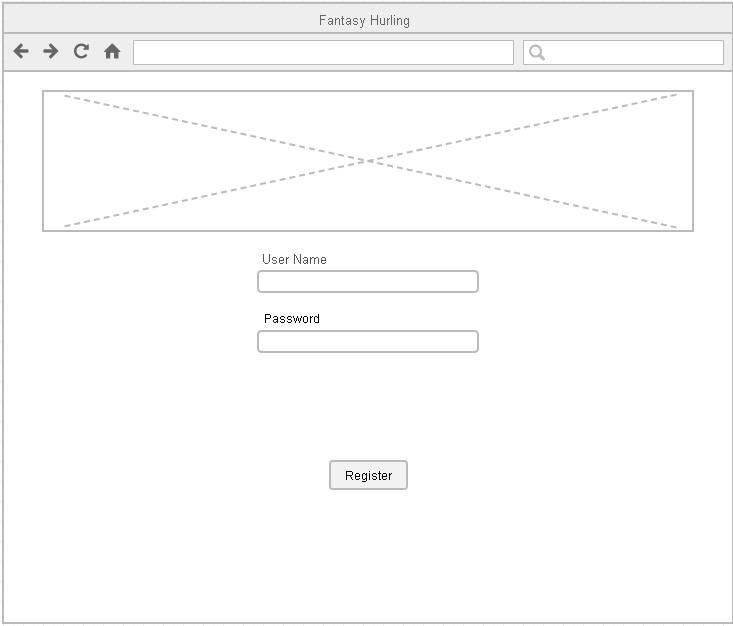


### User makes a transfer



## **Wire framing**

### Login Page

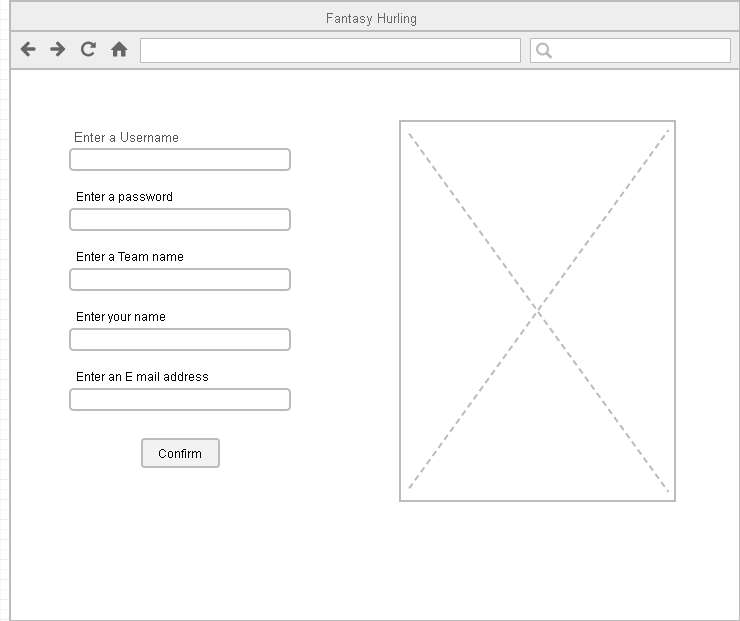


On this page the user can log into the system. They enter a username and password and are logged in. There is an image also.

Elements needed:

* 2 Text boxes
* One button
* One image

### Registration Page

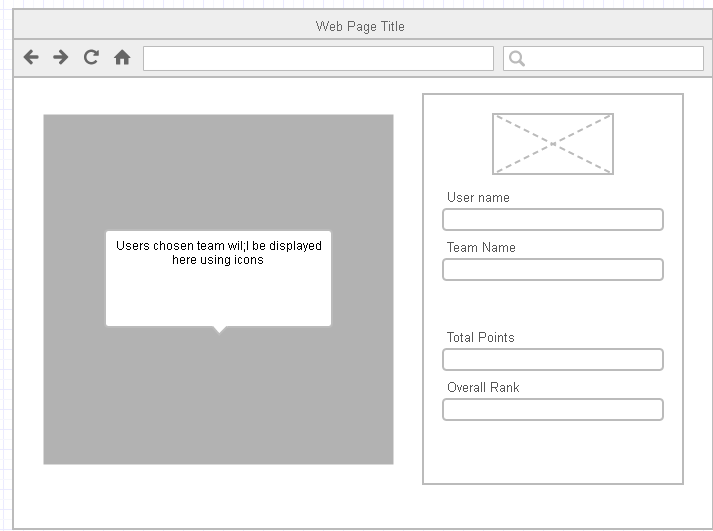


On this page the user can register if they have not already.

Elements needed:

* 5 Text boxes
* One button
* One image

### Team statistics page

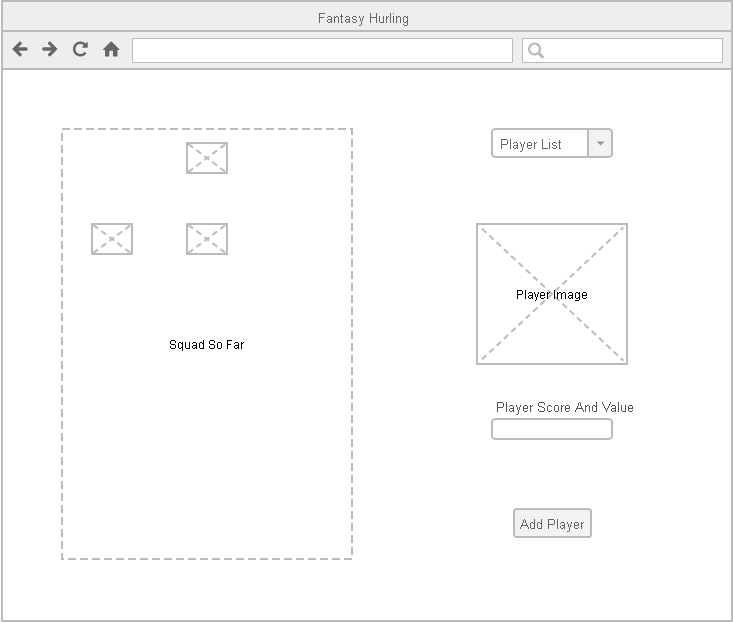


On this page the user can view their team’s current standing and overall points. It shows the user name and team name and the team the user has selected on the left

Elements needed:

* One image
* Team box made up of images and text
* Some basic Divs to show some information based on the user

### Player transfer page

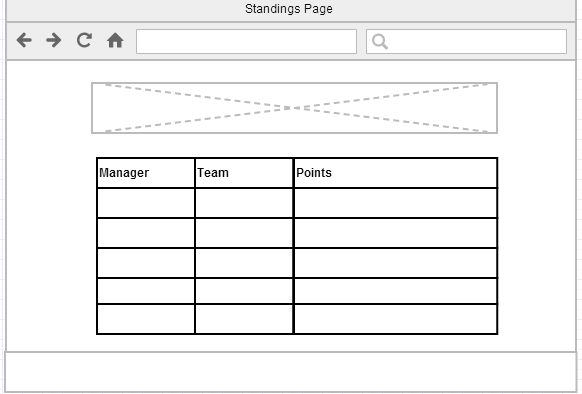


On this page the user can make transfers if they want to. They can search through the database of players, remove a player from their team, and see player score and value. Nothing is confirmed till the user clicks the add player button.

Elements needed:

* One combo box
* Basic Div. to show stats about the player
* One image to show player picture
* One button to confirm change
* A box on the left with all the players selectable and removable from the players team

### Standings Page

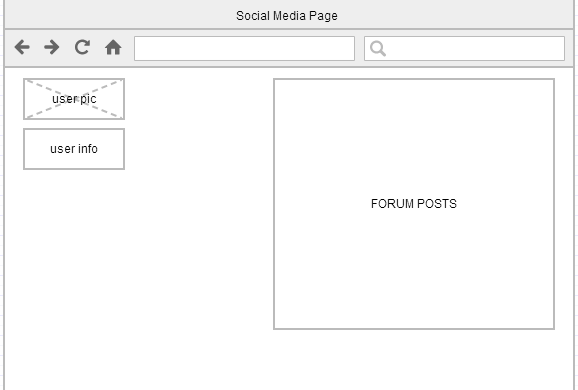


On this page the player can view the top players in the game by points scored. He may be in this list, it will be a simple SQL query displayed in a HTML table.

Elements Needed:

* HTML table
* One image
* Header and footer ECT

### Social Media area



On this main social media page the user can view all recent forum posts and has some information about their profile displayed also, when the user clicks a post, which will be displayed as a title with a link they are brought to a separate page with that post and all replies in it. Here the user can reply to the post also.



Elements needed:

* One image
* One text area
* An area to store forum posts
* A button to post the message

## **Work Flow Plan**

* Due to change of SDLC methodology all DEVELOPMENT and QUALITY ASSURANCE items are merged into PROTOTYPING ITERATIONS item.
* All DESIGN items are reduced to PROTOTYPE DESIGNS (user interface prototype design etc.)

**Individual Task Assignment?**

**David Kelly**

* Front-End design ***responsibility*** (wire frames, photo shop, HTML5, CSS3)
* Documentation involvement
* Testing involvement

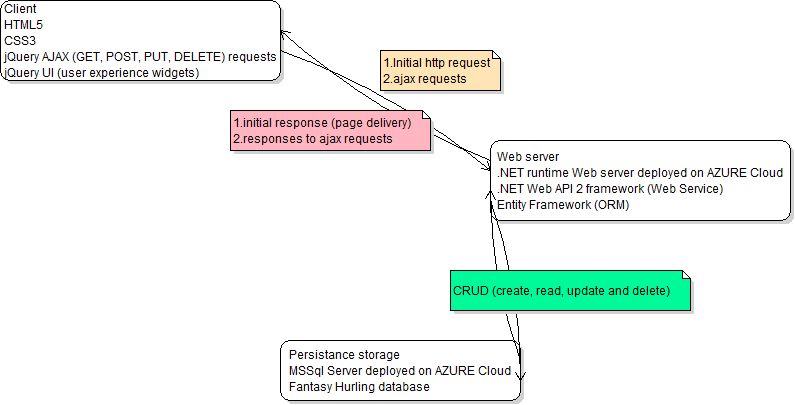
**Michael James**

* Front-End development ***responsibility*** (HTML5, CSS3, vanilla JavaScript, JQuery, JQuery UI)
* Documentation involvement
* Testing involvement

**Martin Zuber**

* Back-End ***responsibility***(MySQL database, .NET Web API 2, Entity Framework ORM mapper, Azure Deployment)
* Documentation involvement
* Testing involvement

## **DESIGN and TECHNOLOGY details**



*Figure 3 Web Application in detail*

**GIT and Github**

We are developing this application using Git distributed revision control system.

Fantasy hurling upstream repository is available at:

<https://github.com/Michaelcj10/Fantasy_Hurling>

Fantasy hurling repository clone is maintained and available at:

<https://github.com/zubidlo/group_project>

Web API prototype upstream repository is available at:

<https://github.com/zubidlo/FanHurApi>

These two repositories are going to get merged in later state of development.

**Front-End Research**

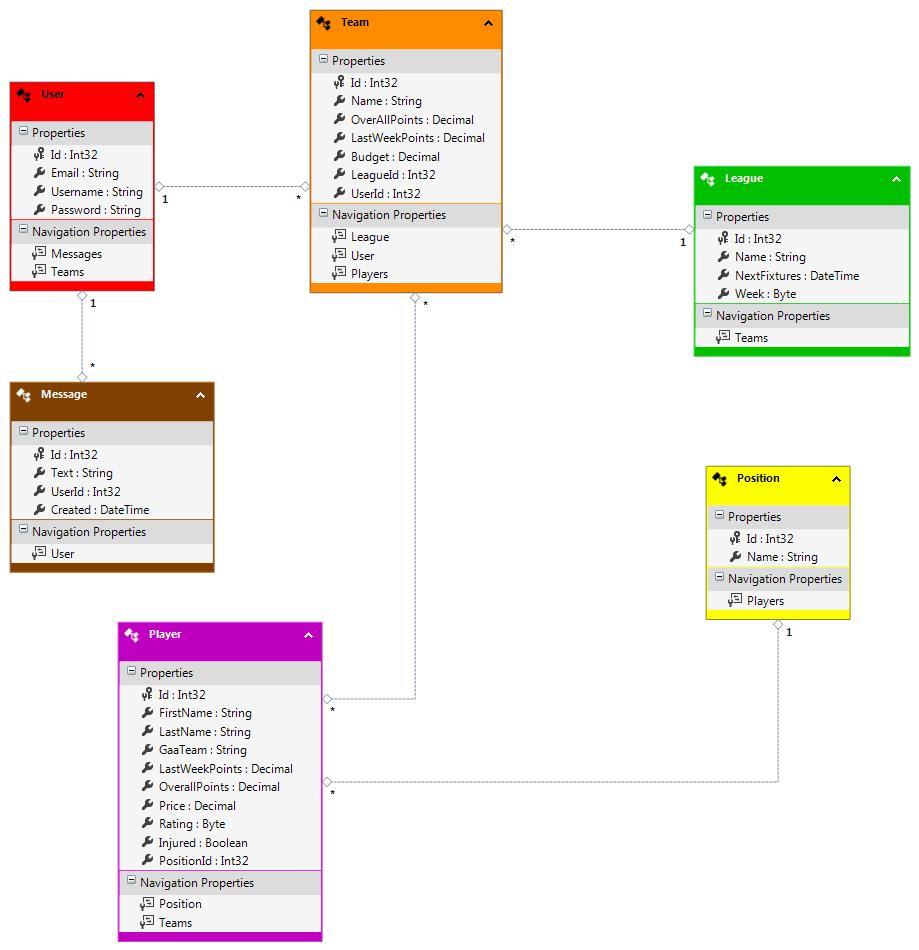
First front end prototype is simple web page using HTML5 ***[2]*** and CSS3 ***[3].***AJAX ***[4]*** request are made through simple web forms using JQuery ***[5].*** In prototyping iteration this web page will gradually turned to user friendly application using JQuery UI ***[6]*** widgets and methods.

Front-End development will be done using IDE. Some recommended options are Sublime Text 2 ***[7]*** or Aptana Studio ***[8].***

**Data Store**

We store Fantasy Hurling data in relational database tables. Initial prototype is design to mimic football like sport team and player statistics and it will gradually mutate into final ‘hurling’ state during prototyping iterations.

*Figure 4 Fantasy Hurling database prototype*



Database will be deployed on Azure Data Store ***[9].*** At the end of development data store gets migrated to Azure Europe North Datacenter located in Dublin ***[10].***

**Back End Research**

Extensive research was done on web application back end development and we decided to build Web API Service ***[11].*** Initial idea was to use Node.js framework ***[12]*** to build the API, but after additional research a decision was made to use ASP.NET Web API ***[13]*** technology. Some of the reasons for this change include:

* Complexity of task versus maturity of Node.js technology
* Potential lack of documentation if a problem in code arises
* Lack of quality connectors to relational databases, Mongo DB (json based object database) is preferred with Node.js
* Additional frameworks to learn building Web API (express framework and more)
* Visual Studio 2013 Community edition free for use now
* 10 web servers free on Azure Cloud with Visual Studio installation
* Visual Studio fully implements Azure storage and web server deployment (on click in a menu and application and database is up on cloud)
* Comprehensive documentation and tutorials for all required frameworks and technologies.
* Technical support and huge community of developers if a coding problem arise.

I must add here that we expect a usage of additional tools and technologies during the project development as a need arises.

## Implementation

Front End Implementation

//put all the crap about the web site here..

Back end implementation

// all the back end stuff here

Code listing

// list the code here? Snippets maybe too much to list…?

Testing

// stuff about testing, unit testing and human testing

Optimization

// section about how we improved it, martin, talk about storing it on Dublin azure servers so it’s quicker

//finding bugs, make them work better…list all that crap

// deployment

Stuff about sticking it online, give the url and git repo url stuff like that

/admin

Talk about how to admin the game, the admin pages that were built…

## **References**

1. What is Prototype model- advantages, disadvantages and when to use it? 2014. What is Prototype model- advantages, disadvantages and when to use it? [ONLINE] Available at: <http://istqbexamcertification.com/what-is-prototype-model-advantages-disadvantages-and-when-to-use-it/>. [Accessed 14 December 2014].
2. HTML5. 2014. HTML5. [ONLINE] Available at: <http://www.w3.org/TR/html5/>. [Accessed 14 December 2014].
3. Introduction to CSS3. 2014. Introduction to CSS3. [ONLINE] Available at:<http://www.w3.org/TR/2001/WD-css3-roadmap-20010523/>. [Accessed 14 December 2014].
4. Ajax | MDN. 2014. Ajax | MDN. [ONLINE] Available at: http. [Accessed 14 December 2014].
5. JQuery. 2014. JQuery. [ONLINE] Available at: <http://jquery.com/>. [Accessed 14 December 2014].
6. Menu | JQuery UI. 2014. Menu | JQuery UI. [ONLINE] Available at:<http://jqueryui.com/m/>. [Accessed 14 December 2014].
7. Sublime Text: The text editor you'll fall in love with. 2014. Sublime Text: The text editor you'll fall in love with. [ONLINE] Available at:<http://www.sublimetext.com/>. [Accessed 14 December 2014].
8. Aptana. 2014. Aptana. [ONLINE] Available at: <http://www.aptana.com/>. [Accessed 14 December 2014].
9. Storage Documentation | Azure. 2014. Storage Documentation | Azure. [ONLINE] Available at: <http://azure.microsoft.com/en-us/documentation/services/storage/>. [Accessed 14 December 2014].
10. Azure Regions. 2014. Azure Regions. [ONLINE] Available at:<http://azure.microsoft.com/en-us/regions/>. [Accessed 14 December 2014].
11. Web API - Wikipedia, the free encyclopedia. 2014. Web API - Wikipedia, the free encyclopedia. [ONLINE] Available at:<http://en.wikipedia.org/wiki/Web_API>. [Accessed 14 December 2014].
12. node.js. 2014. node.js. [ONLINE] Available at: <http://nodejs.org/>. [Accessed 14 December 2014].
13. ASP.NET Web API | the ASP.NET Site. 2014. ASP.NET Web API | the ASP.NET Site. [ONLINE] Available at: <http://www.asp.net/web-api>. [Accessed 14 December 2014].
14. Cross-origin resource sharing - Wikipedia, the free encyclopedia. [ONLINE] Available at: <http://en.wikipedia.org/wiki/Cross-origin_resource_sharing>. [Accessed 14 December 2014].
15. Entity Framework. 2014. Entity Framework. [ONLINE] Available at:<http://msdn.microsoft.com/en-ie/data/ef.aspx>. [Accessed 14 December 2014].