CO3015 Computer Science Project

DOTA2 Web and Android Assistant

Dissertation

Matthew Bennett

May 2015

Department of Computer Science, University of Leicester

Table Of Contents

DECLARATION

All sentences or passages quoted in this report, or computer code of any form whatsoever used and/or submitted at any stages, which are taken from other people's work have been specifically acknowledged by clear citation of the source, specifying author, work, date and page(s).

Any part of my own written work, or software coding, which is substantially based upon other people's work, is duly accompanied by clear citation of the source, specifying author, work, date and page(s).

I understand that failure to do this amounts to plagiarism and will be consIDered grounds for failure in this module and the degree examination as a whole.

Name: Matthew Bennett

Signed: 

Date: 7/5/2015

1 - Abstract

The main aim of this project is to design a system that will assist players of the popular video game Defence of the Ancients 2 (DOTA 2)[1]. The system will enable players to practice and develop their skills at selecting team members, known as 'heroes', in order to provide the team, of which they are part, with the optimum chance of winning any game in which they compete. An artificial intelligence (AI) system will also be designed in order to improve a players hero selection capacity. The system will have multiple levels of difficulty in order to allow every caliber of player to improve their selection skills whilst being challenged. The system will also contain advanced statistical pages providing information on heroes in the game which can be used for user reference and research.

In order to be accessible to the target audience, which is the DOTA 2 player base, the system will be implemented as a web application as well as on the android platform. The system will, in my opinion, prove a useful and engaging tool for all levels of DOTA 2 players.

Throughout the project I will research and discuss the following:

- Functional and non functional requirements that the system is required to meet in order to fulfil the aims and objectives
- The design goals which will enable the functional and non functional requirements to be met
- Implementation of the design goals
- Testing of the system

The project will be subject to critical appraisal as I analyse whether the system has met my initial aims and objectives. I will also consider the social, commercial and economic impact of the system.

Finally, the project will provide me with an opportunity to vastly enhance not only my software development skills but also my project planning and programming skills. I will be required to consider both the issues and risks that may occur during the development of the system and will thus develop strategies in order to manage them. I will develop my ability, knowledge and understanding in numerous technologies, for example Java, MySQL and JSP as I carry out my research.

2 - Introduction

This project is a web and android based assistant application for DOTA 2, a popular multiplayer online battle arena game which is made by the Valve Corporation[2]. The application aims to help players make decisions and hone their knowledge and ability, in order to improve their likelihood of success inside DOTA 2.

Each individual DOTA 2 game enables two teams, a red team and a green team, to play against each other. Each team consists of five individual players, who each select or as it is termed in the game 'draft' a character, as whom they would like to play. Each character is called a 'hero'[3] and each hero possesses its own individual skill set. The system designed intends to develop an assistant to use during the aforementioned drafts, the concept being that when a player selects a hero it can be entered into the assistant and the assistant will provide suggestions on which accompanying heroes should picked or banned (picked or banned are the terms used for selecting or dismissing from the range of heroes available from which each player will chose).

The system will use the win rates between heroes (in previously occurring matches) in order to calculate suggestions of possible heroes which should they be selected together, would provide the team with the optimum chance of winning the game.

The project aims to enable players to practice and develop their skills at drafting heroes whilst playing the 'Captains Mode' game type[4]. Captains Mode is one of a number of different games that can be played within DOTA 2. A Captains Mode draft is the way in which each team chooses which heroes they will play with whilst playing a Captains Mode game and it is the way in which all professional games are conducted as well as a large amount of public games. An example of a pick order is shown below where the green team, who are named the Radiant, picks before the red team, who are named the Dire, because they likely won a coin toss:

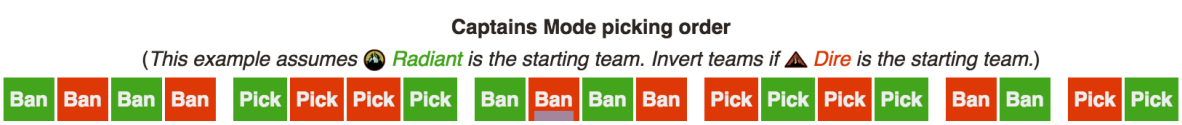


Figure 1.1[4]

A drafting game will also be part of the system. This game is designed to allow users to simulate a draft similar to the one explained above, against an artificial intelligence (AI) system in order to improve their drafting ability. The system will have multiple levels of difficulty in order to allow every calibre of player to improve whilst being challenged. The system will also contain advanced statistical pages providing information on heroes in the game which can be used for user reference and research.

The drafting assistant, one of the main components of my application, helps a user follow a real time Captain's Mode draft, a system in the game that coordinates hero selection, and receive suggestions on heroes to pick and ban in order to out draft an opponent. The drafting game, another large component of my application, is designed to allow a user to play against a computer based drafter. This component will help a user improve naturally at drafting and measure how skilled they are when taking part in a draft where thousands of possible situations can appear. The system uses Valves API[5] (Application Programming Interface) to analyse recent games that have been played in order to generate hero win rates in a multitude of different situations. It uses this data in order to make decisions on which heroes to pick and ban in all situations, both advising the user in the drafting assistant and playing against the user in the drafting game.

The objectives and aims of the system I have worked towards changed throughout the project as I discovered and considered the aspects of each aim and the accompanying complexities. The concept of a draft game was an optional objective in the project plan. After careful consideration I decided the addition of a game would add a significant amount of complexity and useful functionality to my application. I removed the 'smart' log in system following an evaluation of the objective as I decided the complexity and functionality it provided was negated by the time investment required to implement.

The project is intended to meet a demand from the DOTA 2 community that is not currently satisfied as it provides a way in which a user can practice drafting outside of playing the game. The application allows users to improve their understanding of the Captains draft system as well as gathering expertise in choosing heroes to pick or ban.

3 - Requirements

In the following section we will discuss the functional[6] and non-functional[7] requirements that the system should meet in order to fulfil the aims and objectives set out in the projects inception. Firstly we will discuss functional requirements where we will detail the systems high level functional requirements, as well as detailed descriptions. Secondly we will discuss relevant non-functional requirements.

3.1 - Functional Requirements

3.1.1 - High Level

Drafting Assistant

- Allows the user to input up to 5 heroes for the friendly team
- Allows the user to input up to 5 heroes for the friendly bans
- Allows the user to input up to 5 heroes for the enemy team
- Allows the user to input up to 5 heroes for the enemy bans
- Displays all selected heroes using hero avatar images
- Populates a table to show advised picks based on both the friendly and enemy teams
- Populates a table to show advised bans based on both the friendly and enemy teams

Drafting Game

- Provides the user with a choice of three difficulties of AI to draft against
- Allows the user to enter bans and picks following the standard order for a team going first
- As the user bans or picks heroes the AI bans or picks back based on the heroes the user has chosen
- When the user has picked their last hero the game displays a result both as a boolean function of whether the user would win the game on average whilst also indicating the percentage chance the user would have to win the game

3.1.2 - Detailed Requirements

Drafting Assistant

A user should be able to input a set of heroes chosen at any point during a Captains Mode draft

- The system should alert the user if the same hero has been input twice and revert the drafting assistant to the previous stage
- The system removes already selected heroes from the users potential options, so that the same hero cannot be entered multiple times
- When the form is submitted the selected heroes images should be shown
- When the form is submitted the suggested picks and bans should be shown
- Once submitted the hero selection should be saved so that the user can continue selecting heroes as the draft goes on

The system should display suggested picks based on currently selected heroes

- The system confirms that the heroes selected by the user are being correctly passed to the method
- The system gathers data on the selected heroes and returns a set of advised picks, based on which heroes counter the opponents team and synergise with the users team
- The system displays the top ten heroes from the list that have not already been banned or picked

The system should display suggested bans based on currently selected heroes

- The system confirms that the heroes selected by the user are being correctly passed to the method
- The system gathers data on the selected heroes and returns a set of advised bans, based on which heroes counter the users team and synergise with the opponents team
- The system displays the top ten heroes from the list that have not already been banned or picked

Drafting Game

A user is asked to select a difficulty

- The system confirms the selected difficulty and navigates to the respective page

A user should be able to enter a set of picks and bans in the standard Captains Mode drafting order

- The system confirms that the heroes selected are being correctly passed to the method
- Dependant on the difficulty selected the AI chooses the appropriate counter picks based on the users picks, the previous AI picks and heroes average win rates
- The system removes already selected heroes from the users potential options, so that the same hero cannot be entered multiple times

- Once the user has selected their five heroes the result and percentage chance are displayed

3.2 - Non-functional Requirements

Modifiability

The systems three tier architecture style will allow for high levels of modifiability due to the ability to modify any one tier i.e. the logic layer, without having to make significant changes to the other layer. It will be modifiable due to the following criteria:

Suitable Structure -

- With well structured Java classes it will be easy to modify certain methods and understand the impact of the modification, allowing for relatively simple changes and additions to be made
- Well structured web pages will allow for additional front end components to be added easily without major disturbance to the pre-existing pages

Portability

Portability is a key part of the system as it will be run on two separate platforms and certain functionality will be ported between one and the other. The portability of the system will be defined by matching the following criteria:

Functions being portable -

- Functions being portable between different systems and not being heavily reliant on certain packages or methods

User Interface being portable -

- Aspects of the user interface being portable between different technologies and systems

Reliability

The system needs to be reliable as users of the system, DOTA 2 players, play at all times of the day and from all corners of the world. The application needs to be constantly accessible as downtime could severely damage the functionality and reputation of the product. The system will strive to be reliable through the following criteria:

Constant Uptime -

- The system needs to be available for use by any party at all times
- The required data, stored in the database, needs to be available and accessible so the functions of the system are accurate and useful

Up-to-date Data -

- The data stored in my database needs to be relevant for the current version of DOTA 2
- The data needs to be based on recent games played in order to stay reliable and relevant

Usability

The usability of the system is one of the most important features as the system is designed to be used by a wide variety of users. It needs to be logical, easy to navigate and usable to its maximum capability. The system should meet the following criteria in order to achieve these aims:

Easily Navigated -

- The system should be easy to navigate due to a static menu bar with links to all components
- Using the Drafting Assistant and Drafting Game should be straightforward, after each user input the next expected step should be obvious

Comfortable UI -

- The User Interface on both the web and android applications should feel natural and easy to both use and navigate using the respective means, i.e. a standard mouse and keyboard or a touch screen phone/tablet

Testability

The system needs to be highly testable due to the nature of the data, the algorithms used to calculate advised heroes need to be visibly utilising the data correctly. It is also important that the drafting game is testable as we need to be able to produce results clearly in order to show the difference between the three difficulties which are easy, medium and hard. In order to make the system testable the following criteria should be met:

Hero Selection Testing -

- The methods that select advised heroes based on particular inputs need to be tested to prove they are using the database correctly to produce accurate and useful results
- The more complex algorithms for selecting heroes need to be tested to validate the way they use different variables to select one particular hero is as planned

Draft Game Difficulty Testing -

- We need to be able to produce conclusive results, through multiple avenues of testing, that the different difficulty settings inside the Drafting Game are explicitly defined and respectively challenging

4 - Design

In the following section we will discuss the design goals of the system that are required in order to meet the functional and non-functional requirements. We will discuss the software architecture of the system, the hardware & software requirements, data structures used in the system and the algorithms the system uses.

4.1 - Software Architecture

The decision to use a three tier architecture[8] system was due to the suitability of the systems requirements. Due to the nature of the project three explicit tiers were required to store the data, manipulate the data and display the data. A two tier architecture could have been used but would have caused bottlenecks, due to the complexity of the logic tier. Having clear tiers would increase the modifiability of the system and allow easier iterative development. The architecture would also make it easier to port large parts of the system over to other platforms, such as android, something which is vital to the project. Below is a visual representation of the systems architecture.

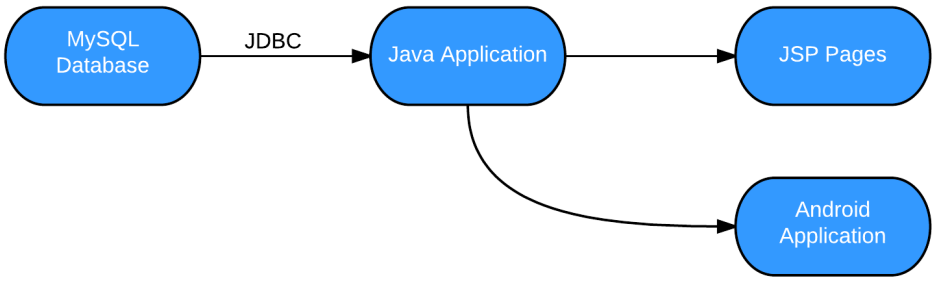


Figure 4.1

4.2 - Data Layer

The Data layer will be made up of a MySQL database with multiple tables. It will be used to store a large amount of records from parsed games that will be parsed as quickly as possible. Therefore the technology needed to be able to hold large integers and be editable quickly. MySQL was, in my opinion, a fairly obvious choice for the following reasons[9][10]:

- Accessibility - Its available for free download and is widely installed and used on many operating systems and platforms

- Ease of Use - I was already familiar with the technology and was comfortable with implementing and managing multiple tables and databases
- Speed - Its a minimalistic tool and would provide all of the features I required without any additional features that may cause bloating or slower editing speeds
- Size - It can handle massive amounts of data, up to 50 million rows, and no matter how much the system was scaled the technology should never become an issue
- Reliability - MySQL has built in reliability measures that prevent the loss of data or streams during crashes

Less than 1.5 billion DOTA 2 games have been played[11], in my opinion, the figure is probably significantly less. The maximum, at this moment in time, any cell could therefore hold 1.5 billion. As the maximum length of an integer in MySQL is just over 2 billion[12], meaning it is more than suitable for the records to be stored as integers. The memory size an integer takes in MySQL is 4 bytes[11]; in a default MySQL database, with a 4 GB limit[13], just over 1 billion integers could be held. Due to the fact that the max hero ID is currently 111 in DOTA 2[2], the maximum table size required would be 111^2 (12,321). Therefore I would need to populate 87,000 tables in order to have memory problems, something that, in my opinion, would never happen. I predict that there will never be a risk of memory overflow using a standard MySQL database server.

The main objectives of the MySQL database will be to store the following:

- How often heroes beat other heroes
- How often heroes win alongside other heroes
- How often heroes win on average

In order to achieve these objectives the following tables will be deployed:

hero_matchups

hero_ID	int
ID_1	int
ID_2	int
…	
ID_111	int

This table will be used to track how many times each hero has beaten each of the other heroes. This will allow the system to calculate hero versus hero win rates by diving the amount of times one hero has beat another by the total amount of games they have played against each other.

hero_synergy

hero_ID	int
ID_1	int
ID_2	int
…	
ID_111	int

This table will track the amount of times a hero has played alongside another hero and won.

hero_countersynergy

hero_ID	int
ID_1	int
ID_2	int
…	
ID_111	int

This table will be used to track how many times a hero has played alongside another hero and lost. In conjunction with the hero_synergy table the system will be able to calculate the percentage of games two heroes win when they play together.

hero_average

hero_ID	int
win_rate	int

This table will be used to track every heroes average win rate.

day_matchups1

hero_ID	int
ID_1	int
ID_2	int
…	
ID_111	int

I will have multiple tables similar to this (day_matchups1, day_matchups2 ect.) which will inform my statistics that require a change in hero win rate over a given time. They will hold similar data to that of hero_matchup but over a limited period of time.

4.3 - Logic Layer

My logic layer is required to access my Data layer and manipulate data from it in order to produce a dynamic and interactive front end. I decided to write the majority of my logic in Java for the following reasons[14]:

- Portability - Due to the nature of Android development, which is mainly written in Java, I knew that using Java for the initial web application would make the system vastly portable to the platform I planned to use
- Support - There are extensive resources and interfaces for Java which allow easy research and development of unknown areas of the language
- Database Connection - Java has a popular and well documented driver to connect to MySQL Databases, JDBC. I am familiar with the use of this driver and have experience in its implementation

4.4 - Presentation Layer

I decided I wanted the main user interface to be web based so that it could be accessed by the majority of the proposed client base. Due to the fact web based applications can be used on any platform with a web browser it has a much lower barrier to its use than other front end technologies.

The presentation layer was required to be dynamic, receive user input and change based on the information provided. It also needed to be able to access the Java classes in order to extract data from the database and also from certain calculation methods inside my logic layer. Therefore Java Servlet Pages (JSP's) were selected due to the following factors[15]:

- Compatibility - JSP's are a relatively simple way to implement a dynamic web based user interface that can access and use Java functionality
- Modifiability - JSP's can be edited and modified without the need to re-write or change large amounts of logic, they can be written as simple html pages and then have the logic aspects inserted
- Complexity - Due to the fact that fairly complex log and data processing is required a logic layer is necessary. Many dynamic web languages are implemented by combining the logic and presentation layers, this could have created bottlenecks in the system, thus reducing performance. A JSP presentation layer can be used in partnership with a large Java based logic layer, increasing modifiability, performance and portability

In terms of aesthetics I decided to implement the bootstrap[16] platform for css. This is a free package of css tools and allows easy access to attractive styles which are fit for both full size screens and mobile devices.

4.5 - Hardware and Software Requirements

The system requires certain elements of hardware and software in order for it to be fully implemented. As the system is highly data reliant, it is vital that the data can be reached and processed in order to provide functionality to the user.

4.5.1 - Hardware

The system requires certain pieces of hardware in order to be implemented fully. These pieces of hardware will make up the three tier system and ensuring they can communicate is essential to the system meetings it's requirements.

A MySQL server is needed to store the data that allows my application to make informed suggestions to the user. It will contain data gathered from Valve's API. Using a MySQL server will ensure all of the data the system requires is accessible.

An apache tomcat based web server will be required in order to host the web based application. It will display the front end JSP pages of the application and allow the user to interact with the system.

4.5.2 - Software

During the development of the system various pieces of software will be required. These software pieces will allow direct communication with the server, allowing maintenance and alterations to occur if deemed necessary.

Most notably the development environment Eclipse with the Java EE plugin will allow the system to be hosted on a local apache server. This will enable testing of the methods and functions during development. Eclipse will allow the system to be developed in a realistic environment that prepares it for hosting on a web server.

MySQL Workbench has allowed interaction and manipulation of the database in order to test and manage the way my data is being parsed and then stored. It will allow me to export the database to host on my MySQL server during implementation.

4.6 - Data Structures

In order for the logic layer to extract and process data from the database it is required to utilise complex data types. As the system will need to take into consideration over one hundred heroes with multiple win rates the correct data structures need to be used. It will need to be able to manipulate and extract the correct pieces of data from these structures and therefore the methods need to utilise these structures effectively.

In order to store a certain hero, or set of heroes win rates versus all other heroes the system will use two dimensional arrays[17]. These two dimensional arrays will need to hold multiple arrays of length two that hold both each heroes ID and the relevant win rate for that hero. The main reason this data type is appropriate is due to the ability to index arrays, take a particular entry at any point in the array and manipulate it. This allows the system to calculate a particular hero win rate from any array, a feature that will be required whilst implementing the requirements.

4.7 - Algorithms

The logic layer will need to incorporate a significant number of algorithms in order to meet the systems requirements. The following algorithms will need to be defined:

- Calculating the suggested picks in the drafting assistant at any point in a draft
- Calculating the suggested bans in the drafting assistant at any point in a draft
- Calculating a pick or ban in the easy difficulty drafting game at any point in a draft
- Calculating a pick or ban in the medium difficulty drafting game at any point in a draft
- Calculating a pick or ban in the hard difficulty drafting game at any point in a draft

The system will display the best ten picks and bans during any drafting situation in the drafting assistant, therefore the first two algorithms that are defined need to gather the correct information from the database and process it into a list of ten hero IDs. The algorithm will be defined as a method that will take the enemy teams heroes in order to output picks and the friendly teams heroes in order to output bans. This is because the best way to draft is by picking heroes that counter the opponent and banning heroes that counter you. The method will gather the two dimensional arrays for all heroes entered and combine them into one, two dimensional array via mean averaging. It will then sort the array and output the top ten heroes by percentage.

The easy difficulty drafting AI needs to be relatively un-intelligent and should be easily beaten by someone with substantial knowledge of DOTA 2. It therefore will gather the worst thirty heroes to pick or ban in any situation and pick a random hero from that array. Having the easy drafting system contain a random element will allow it to remain interesting and engaging, by removing the scenario of the same draft happening over and over again.

The medium difficulty AI needs to be able to challenge a user with significant DOTA 2 knowledge whilst being beatable. Therefore the method will collect the top ten heroes to pick or ban in any situation and pick a random hero from that array. This will make the difficulty engaging and challenging but allow the user to overcome the system with good decisions alongside a bit of luck.

The systems hard drafting AI needs to be difficult enough to challenge even the highest calibre of DOTA 2 drafters. Due to this it will need to take into account multiple factors when deciding the correct hero to chose. The method will take the best ten hero choices for any draft situation and the ten heroes with the highest average win rate. It will then choose

the first hero from the first list that also appears in the second, if no heroes appear in both lists it will pick the first hero from the first list. This allows the system to not only think about the current situation but also future decisions the user may make.

4.8 - Parser

The XML parser needs to be able to traverse an XML document output by Valve's API and collect the following data:

- Which team won
- Which hero was on each team

It then needs to use the data in order to fill multiple tables in the database, namely hero_matchups, hero_synergy and hero_countersynergy. The parser will take all of the heroes who belonged to the winning team and increment the correct cells, in hero_matchups, which represent the amount of wins against each of the enemy teams heroes. It will also increment the correct cells, in hero_synergy and hero_countersynergy, that represent each winning hero winning alongside each other winning hero and visa versa for the loosing team.

5 - Implementation

In the following section we will discuss how the design goals and requirements detailed above were implemented. I will discuss how I have implemented the systems MySQL Database and its main Java classes, namely my XML Parser, Drafting Assistant and Drafting Game.

5.1 - MySQL Database

The MySQL database that holds all of the data for the system to function contains a number of tables that are all required to gather and process the systems suggestions.

The hero_matchup table contains data on how frequently each hero has beat each other hero. This table is generated by the following statement:

```
CREATE TABLE hero_matchups (
hero_ID int(3) NOT NULL,
ID1 int(6) NOT NULL,
ID2 int(6) NOT NULL,
ID3 int(6) NOT NULL,
…
ID111 int(6) NOT NULL,
PRIMARY KEY(hero_ID)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```

The hero_synergy and hero_countersynergy tables hold data on how frequently heroes have won whilst playing alongside each other and how often heroes have lost playing alongside each other respectively. They are both generated by similar create statements, one of which is shown here:

```
CREATE TABLE hero_synergy (
hero_ID int(3) NOT NULL,
ID1 int(6) NOT NULL,
ID2 int(6) NOT NULL,
ID3 int(6) NOT NULL,
…
ID111 int(6) NOT NULL,
PRIMARY KEY(hero_ID)
```

```
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```

The hero_average table holds each heroes average win rate calculated using the hero_matchup table. It is created by the following statement:

```
CREATE TABLE hero_average (
hero_ID int(6) NOT NULL,
win_rate int(6) NOT NULL)
Engine=InnoDB DEFAULT CHARSET=latin1;
```

The day_matchups1 table, and those like it, will hold data similar to that of the hero_matchups table just restricted over a period of time. They will be created by statements similar to the following:

```
CREATE TABLE day_matchups1 (
hero_ID int(3) NOT NULL,
ID1 int(6) NOT NULL,
ID2 int(6) NOT NULL,
ID3 int(6) NOT NULL,
…
ID111 int(6) NOT NULL,
PRIMARY KEY(hero_ID)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```

5.2 - Data Parser

```
public static voID beat(String won, String lost){
      if (!won.equals("0") && !lost.equals("0") && !won.equals(null)
      && !lost.equals(null)){
      databaseconnector("UPDATE hero_matchups SET ID" + lost + "=ID"
      + lost + "+1 where hero_ID =" + won);
```

The beat method is expected to be used when one hero, the first ID input, has beaten another hero, the second. The method takes two hero IDs as strings and checks neither of them are null or zero. It then uses the Database Connector method to perform the appropriate MySQL statement. The statement will increment a cell, in the hero_matchups table, in the row of the winning hero and column of the losing hero.

```
public static voID wonwith(String hero1, String hero2){
      if (!hero1.equals("0") && !hero2.equals("0") && !hero1.equals(null) &
      & !hero2.equals(null)){
      databaseconnector("UPDATE hero_synergy SET ID" + hero2 + "=ID" +
      hero2 + "+1 where hero_ID =" + hero1);
```

The wonwith method is expected to be used when one hero, the first ID input, has won a game with another hero, the second. The method takes two hero IDs as strings and checks neither of them are null or zero. It then uses the Database Connector method to perform the appropriate MySQL statement. The statement will increment a cell, in the hero_synergy table, in the row of the second hero and column of the first.

```
public static voID lostwith(String hero1, String hero2){
      if (!hero1.equals("0") && !hero2.equals("0") && !hero1.equals(null)
      && !hero2.equals(null)){
      databaseconnector("UPDATE hero_countersynergy SET ID" + hero2 + "=ID"
      + hero2 + "+1 where hero_ID =" + hero1);
```

The lostwith method is expected to be used when one hero, the first ID input, has lost a game with another hero, the second. The method takes two hero IDs as strings and checks that neither of them are null or zero. It then uses the Database Connector method to perform the appropriate MySQL statement. The statement will increment a cell, in the hero_countersynergy table, in the row of the second hero and column of the first.

```
public static voID computewins(String[] Game){
      if (Game[1].equals(“Radiant")){
            for(int i=2; i<7; i++){
            for(int j=7; j<12; j++){ beat(Game[i], Game[j]);}
      else{
            for(int i=7; i<12; i++){
            for(int j=2; j<7; j++){beat(Game[i], Game[j]);
```

The computewins method takes an array of strings which contains game ID, result and the IDs of all heroes involved. It is expected to be used inside parsing methods in order to use the beat, wonwith and lostwith methods in order to correctly update the database after a game is parsed.

```
public static voID parserecentgames(String url){
      DOMParser parser = new DOMParser();
```

```
parser.parse("https://api.steampowered.com" + gameID + "key=");
Document doc = parser.getDocument();
```

The parseparticulargame method utilises the computewins method to parse and process the data we need from an XML page outputted by Valves DOTA 2 API. It takes a game ID as an integer and will run through the XML sheet produced using the above DOM parser code.

```
if(res != null){
            if(res.getTextContent().equals(“true")){ Game[1] = "Radiant";}
            else{ Game[1] = “Dire";}
```

In the above snippet the method checks which team won the game using the radiant_win element. If this element is true, the radiant team won the game, otherwise the dire team won.

```
NodeList players = doc.getElementsByTagName("player");
      int val = 2;
      for (int i=0; i<players.getLength(); i++){
            Node ply = players.item(i);
            NodeList ply1 = ply.getChildNodes();
            for(int j=0; j<ply1.getLength(); j++){
                  Node ply1c = ply1.item(j);
                  if(ply1c.getNodeName() == "hero_ID"){
                        Game[val] = ply1c.getTextContent();
                        val++;}}}
      computewins(Game);
```

It then gets all of the hero IDs in the game in the order of radiant team followed by dire team. It fills a String array with the game ID, result and hero IDs in order to use the computewins method for this game. This is shown in the code snippet shown above.

```
public static voID fillaverages(){
      double[][] averages = new double[111][2];
      for(int i=0; i<111; i++){
            double[][] ch = DatabaseManager.getparticularhero(i+1);
            double ca = 0;
            for(int j=0; j<111; j++){ca = ca + ch[j][0];}
            averages[i][0] = i+1;
            averages[i][1] = (ca/109);}
```

```
for(int k=0; k<111; k++){
      double ID = averages[k][0];
      double wr = averages[k][1];
      databaseconnector("INSERT INTO hero_average (hero_ID,
      win_rate)" + "VALUES(" + ID + ", " + wr + ")");}}
```

The fillaverages method fills the hero_average table with each heroes average win rate using the hero_matchups table data. It uses a method from the Database Manager class called getparticularhero to get a heroes win rates against every other hero in a 2d double array, it then averages the percentages in this array and inserts this average into the hero_average table.

5.3 - Drafting Assistant

My getparticularhero method is designed to take an individual heroes ID and return a 2d double array with every heroes ID and their average 'success score' with the hero submitted. The success score is the mean of that heroes percentage win rate against a hero and win rate with a hero, meaning it calculates heroes that would be good picks or bans. It uses code similar to the following snippet four times to generate arrays. It first generates wins against a hero and losses against a hero, then wins with a hero and losses with a hero.

```
int[][] wins = new int[111][2];
int[] hero = new int[2];
con = DriverManager.getConnection("jdbc:mysql://
mysql.mcscw3.le.ac.uk:3306/mb508", "mb508", "iledstro");
Statement s = con.createStatement();
ResultSet rs = s.executeQuery("SELECT * FROM hero_matchups
                              WHERE hero_ID = " + heroID);
while(rs.next()){
      for(int i = 1; i<112; i++){
            hero[0] = rs.getInt(i+1);
            hero[1] = i;
            wins[i-1][0] = hero[0];
            wins[i-1][1] = hero[1];}}
```

It then uses code as shown in the following snippet to combine two arrays, such as wins against and losses against, to get an array populated with win percentages against each hero.

```
double[][] percentheroes = new double[111][2];
            for(int i=0; i<111; i++){
                  double win1 = wins[i][0];
                  double loss1 = losses[i][0];
                        if((wins[i][0] + losses[i][0] != 0)){
                              percentheroes[i][0] = round
                              (((win1/(win1+loss1))*100),2);
                              percentheroes[i][1] = (i+1);} else{ }}
```

After generating two arrays, win percentage against and win percentage with, it uses similar code to that above to combine these two arrays in order to create the final returned 'success score' array.

```
public static double[][] getworstheroes(double[][] heroarray, int[]
bannedheroes){
```

The getworstheroes method takes a 2d double array and an integer array and proceeds to sorts the double array and returns the bottom 10 heroes in the 2d double array by percentage, who are not in the integer array. It uses the code snippet below to populate an array with the 10 heroes from the sorted double array.

```
double[][] worstheroesarray = new double[10][2];
      for(int i = 0; i<20; i+=2){
            worstheroesarray[(i/2)][0] = worstheroes.get(i);
            worstheroesarray[(i/2)][1] = round(worstheroes.get(i+1),2);
      return worstheroesarray;}


public static double[][] getbestheroes(double[][] heroarray){
```

The getbestheroes method takes a 2d double array and an integer array, it then sorts the double array and returns the top 10 heroes in the sorted double array by percentage, who are not in the integer array. It uses effectively the same method as getworstheroes.

```
public static double[][] getgrouphero(int[] heroIDs, int[] bannedheroIDs){
```

The getgrouphero method takes two integer arrays as parameters, intended to be a list of selected heroes and a list of all heroes currently picked or banned. It's purpose is to generate a

2d double array for between one and five heroes, input via the first array parameter, which shows the worst heroes against those heroes. In order to show suggested bans against any size of enemy team the code snippet below is used to generate the arrays for each hero entered in the first parameter.

```
if(heroIDs[0] != 0){
      hero1 = getparticularhero(heroIDs[0]);
      count++;}
```

It then uses the following snippet to combine these arrays into one array with averages.

```
for(int i = 0; i<110; i++){
      if(i+1 == heroIDs[0] || i+1 == heroIDs[1] || i+1 == heroIDs[2]
      || i+1 == heroIDs[3] || i+1 == heroIDs[4]){
            combinedheroes[i][0] = 100;
            combinedheroes[i][1] = i+1;}
else{
      combinedheroes[i][0] = round(((hero1[i][0] + hero2[i][0] +
      hero3[i][0] + hero4[i][0] + hero5[i][0])/(count)),2);
      combinedheroes[i][1] = i+1;}}
```

```
public static double[][] getgroupheroban(int[] heroIDs, int[]
bannedheroIDs){
```

The getgroupheroban method takes two integer arrays as parameters, intended to be a list of selected heroes and a list of all heroes currently picked or banned. It's purpose is to generate a 2d double array for between one and five heroes, input via the first array parameter, that shows the best heroes against those heroes inputted. In order to show suggested picks against any possible team.

```
public static String getheroname(int heroID){
      String heroname = "";
      switch (heroID){
                  case 1: heroname = "Antimage"; break;
                  case 2: heroname = "Axe"; break;
                  case 3:…
      }
```

```
	return heroname;}
```

The getheroname method takes any defined hero ID as an integer and returns the name of the appropriate hero as a string.

```
public static List<String> getAllHeroes(){
	List<String> allheroes = new ArrayList();
	for(int i = 1; i <= 111; i++){
		if(getheroname(i).equals(“")){ allheroes.add("NOHERO");}

		else{ allheroes.add(getheroname(i));}}
		return allheroes;}
```

The getAllHeroes method uses the getheroname method to produce a List of Strings of all hero names in the game and outputs that list.

```
public static String[] win(int[] fheroIDs, int[] eheroIDs){
	…
	for(int i = 0; i < 5; i++){
		for(int k =0; k <= 110; k++){
			if(hero1[k][1] == eheroIDs[i]){
				total = total + hero1[k][0];}
			if(hero2[k][1] == eheroIDs[i]){
				total = total + hero2[k][0];}
			if(hero3…
				…
			total = total + hero5[k][0];}}}
			perc = round((total)/25,2);
			if(perc >= 50){win = true;}
			result[0] = String.valueOf(win);
			result[1] = String.valueOf(perc);;
			return result;}
```

The method win takes two integer array that are designed to both contain 5 explicit hero IDs. It then calculates which team has the better average win rate against the other i.e. which team should, on average, win the game more. It does this by calculating the win rate of all the friendly heroes against every enemy hero, then if the mean of these numbers is above 50 ‘true’ is returned.

```
public static double[][] getaverages(int[] bans){
      double[][] result = new double[111][2];
      con = DriverManager.getConnection("jdbc:mysql://mysql.mcscw3.le.ac.
                                    uk:3306/mb508", "mb508", "iledstro");
      Statement s = con.createStatement();
      ResultSet rs = s.executeQuery("SELECT win_rate FROM hero_average");
      while(rs.next()){
            for(int i = 1; i<112; i++){
            rs.absolute(i);
            result[i-1][0] = rs.getInt(1);
            result[i-1][1] = i;}}
      return result;}
```

The getaverages method takes an integer array, intended to contain all heroes already banned or picked, and returns a 2d double array of all available heroes and their average win rate. It uses the hero_average table data to populate this array.

5.4 - Drafting Game

```
public static String[] win(int[] fheroIDs, int[] eheroIDs){
```

The win method takes two integer arrays, intended to be filled with the hero IDs of two teams of five. It then calculates which team would win the game on avergage, submitting true if it is the first team, false if it was the second. It also calculates the rate at which the first team would win or loose the game on average. It does this using the following code snippet

```
for(int i = 0; i < 5; i++){
      for(int k =0; k <= 110; k++){
            if(hero1[k][1] == eheroIDs[i]){
                  total = total + hero1[k][0];}
            if(hero2[k][1] == eheroIDs[i]){
                  total = total + hero2[k][0];} …}
      perc = round((total)/25,2);
      if(perc >= 50){win = true;}
      result[0] = String.valueOf(win);
      result[1] = String.valueOf(perc);;
      return result;}
```

```
public static int randomhero(){
	List<String> allheroes = DatabaseManager.getAllHeroes();
	Random random = new Random();
	int heroID = random.nextInt(allheroes.size());
	if(heroID != 0 && heroID != 24 && heroID != 108){return heroID;}
	else{randomhero();}
	return randomhero();
```

The randomhero method fills a list of strings using the allheroes method from the Database Manager class and then selects a random hero ID from this list. It then checks the randomly generated ID is not 0, 24 or 108, as these are all elements inside the allheroes array that do not correspond to heroes. If the ID is 0, 24 or 108 it begins the method again and if it is not it returns the ID. This is the most basic level of my games ability to choose heroes randomly.

```
public static List<Integer> tenrandomheroes(){
		List<Integer> randomheroes = new ArrayList();
		for(int i = 0; i<10; i++){randomheroes.add(randomhero());}
		return randomheroes;}
```

The tenrandomheroes method uses the random hero method to fill a List with ten random hero IDs as integers.

```
public static int randomtop10heroes(int[] heroes, int[] bans){
	int hero = 0;
	double[][] result = DatabaseManager.getgrouphero(heroes, bans);
	for(int i=0; i < bans.length; i++){
		if(bans[i] != 0){
			for(int j=0; j < result.length; j++ ){
				if(result[j][1] == bans[i]){
					result[j][0] = 0;
					result[j][1] = 0;}}}}
	double[][] topten = new double[10][2];
	for(int k=0;k<10;k++){
		topten[k][0] = result[k][0];
		topten[k][1] = result[k][1];}
	Random random = new Random();
	int heroID = random.nextInt(10);
	hero = (int) (topten[heroID][0]);
```

```
	return hero;}
```

The randomtop10heroes method takes two integer arrays filled with hero IDs. It then uses the getgrouphero method to get the best heroes versus the hero IDs in the first array parameter. It proceeds to remove any hero in the second array from the array of advised heroes subsequently taking the top ten heroes from this array of advised heroes and returning the ID of one of them at random.

```
public static int bestaveragehero(int[] bans){
	int hero = 0;
	double[][] result = DatabaseManager.getaverages(bans);
	for(int i=0; i < bans.length; i++){
		if(bans[i] != 0){
			for(int j=0; j < result.length; j++ ){
				if(result[j][1] == bans[i]){
					result[j][0] = 0;
					result[j][1] = 0;}}}}
	for(int k=0; k<111; k++){
		hero = (int) result[0][1];}
	return hero;}
```

The bestaveragehero method takes an integer array filled with hero IDs and returns the hero with the highest win rate on average that is not in the array parameter.

```
public static int randombot30heroes(int[] heroes, int[] bans){
```

The randombot30heroes method takes a particular drafting situation and gets the array for the best hero picks for that situation. It then gets the top ten heroes from this array using the following code snippet

```
double[][] topten = new double[(result.length - 30)][2];
for(int k=result.length - 1;k>30;k--){
	topten[tracker][0] = result[k][0];
	topten[tracker][1] = result[k][1];
	tracker++;}
```

It then takes a random hero ID from this array and returns it using the following code snippet

```
Random random = new Random();
int heroID = random.nextInt(result.length - 30);
while(hero == 0){
      heroID = random.nextInt(result.length - 30);
      hero = (int) (topten[heroID][0]);}
return hero;}
```

```
public static int smartpick(int[] heroes, int[] bans){
```

The smart pick function is the systems most complicated drafting AI algorithm. It gathers two arrays, one filled with the best heroes on average and one with the best heroes for the current drafting situation. It then takes the top ten heroes from each array. It then returns the highest hero which is in both arrays, if there is no one hero in both arrays it returns the top hero in the array that was picked based on the current draft. It does this using the following code snippet

```
hero = (int) (result[0][0]);
for(int i = 0; i < 10; i++){
      for(int j =0; j < 10; j++){
            if(average1[i][1] == result[j][0]){
                  hero = (int) average1[i][1];}}}
return hero;
```

5.5 - Statistics Pages

```
public static double getstandarddev(int heroid){
```

The getstandarddev method takes a hero ID and is intended to return the standard deviation of that heroes average win rate over the time documented in the database. It takes the amount of wins and losses by a hero from certain time based tables using code like the snippet below:

```
ResultSet rs = s.executeQuery("SELECT * FROM hour1_matchups
                              WHERE hero_id = " + heroid);
while(rs.next()){
      for(int i = 1; i<112; i++){
            wins1 = wins1 + rs.getDouble(i);}}

ResultSet rs1 = s.executeQuery("SELECT id" + heroid + "
                               FROM hour1_matchups");
      while(rs1.next()){
            losses1 = losses1 + rs1.getDouble(1);}
```

It then uses these win and loss figures to calculate average hero win rates from each of the specified time based tables. It then uses these figures to calculate the standard deviation of the heroes win rate over the time using the following snippet:

```
double h1wr = (wins1 / wins1 + losses1) * 100;
double h2wr = (wins1 / wins1 + losses1) * 100;
double h3wr = (wins1 / wins1 + losses1) * 100;
double h4wr = (wins1 / wins1 + losses1) * 100;
double mwr = (h1wr + h2wr + h3wr + h4wr)/4;
sd = round(Math.sqrt((h1wr - mwr)*(h1wr - mwr) +
               (h2wr - mwr)*(h2wr - mwr) +
               (h3wr - mwr)*(h3wr - mwr) +
               (h4wr - mwr)*(h4wr - mwr)/4),2);
return sd;}
```

5 - Testing

In the following section we will discuss the testing performed on the system and how it helped me to iterate and improve. It will show progress towards the functional and non-functional requirements via user testing and testing on the draft game difficulties.

5.1 - User Testing

User Testing was gathered by asking DOTA 2 players to use and test the application. Feedback was requested upon the usability, functionality and general opinion of the usefulness of the system. I attempted to gather a wide variety of feedback from people with different levels of knowledge of DOTA 2 for example expert players as well as newcomers to the game. This approach has allowed me to gauge how useful the system is for the entire DOTA 2 community. The results of my user testing are detailed below:

The main changes that resulted from my user testing were the aesthetics of my web application. It changed rather drastically in colour, layout and design as you can see in the differences between figure 5.1 and figure 5.2 below. This was due to users of the system advising that it felt a little 'clunky' when used and could also be improved visually. Below you can see the visual difference before user testing and after.

Figure 5.1

I also received feedback that the way in which heroes are entered in both the drafting assistant and the drafting game could be improved. My users felt that it was tedious to press a submit button after every hero input. An auto submission feature was implemented into my forms ensuring that whenever a hero is entered it automatically processes them into the current draft. This, in my opinion, and the opinion of the users, makes both systems feel much more natural and smooth.

The users found great value in the functionality of the system and plan to continue to use the product to improve their drafting skills and knowledge. All the users claimed that the drafting game would be a fun and useful tool to use in between DOTA 2 games. See the appendix for screenshots of my entire web application.

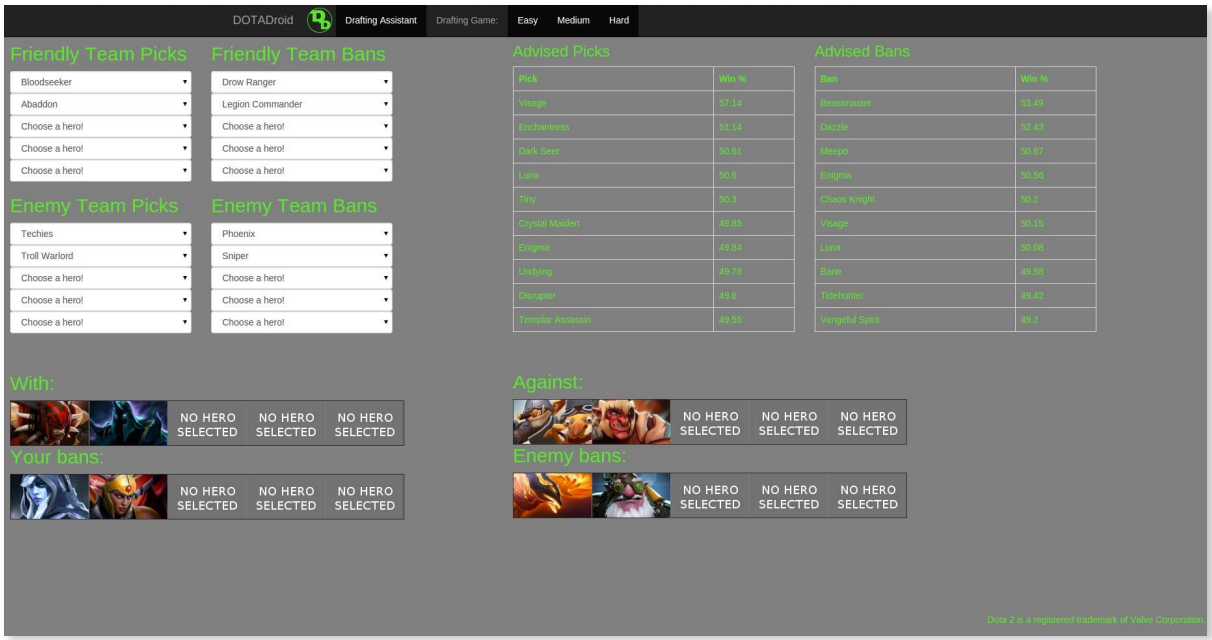


Figure 5.2

5.2 - Draft Game Testing

My draft game required intense testing in order to clarify the differences between each of the difficulties. If the difficulties are not sufficiently separate they may not provide their ultimate purpose of allowing players of different ability levels to challenge themselves and improve. Below you can see the results of the Draft Game testing which clearly show the differences between the difficulties. The below chart shows the win rate of a user picking all random heroes when played against the three difficulties:

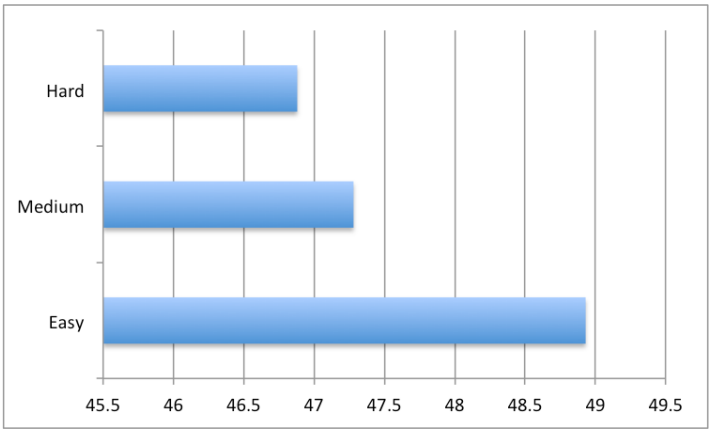


Figure 5.3

As you can see from the above chart, my three difficulties are explicit in difficulty and allow the user to progress through each difficulty as they become a more experienced drafter. The difficulty curve becomes difficult to manipulate after 45% win rate due to the win rates of heroes rarely falling below that figure[18]. Picking the best set of heroes against another set of heroes usually only gives a few percentage points of advantage in a game, therefore a drafting difficulty with a higher than 55% average win rate would be highly problematic.

6 - Critical Appraisal

In the following section I will discuss how well the system meets the requirements laID out during the inception of the project. Analysis of how the system has met certain aims and objectives, and why other aims and objectives may have been excluded from the system will take place. The social, commercial and economic impact of the system will be discussed as well as my own personal development during the development process.

6.1 - Critical Analysis

The aims and objectives I identified for the project during the planning stage have changed drastically. They have been altered and reshaped for many reasons such as improved knowledge, increased understanding of the user and changing requirements. Originally I set out two essential aims, two recommended aims and a large list of optional aims as detailed below:

6.1.1 - Essential Aims

To develop an attractive and functional web application that imports data from Valve's API to power a reliable and helpful drafting assistant

This is a fairly complex aim. It encompasses a large amount of the project. Therefore defining if this aim was met is not an easy task and because of this I will detail each of the its different components.

Web Application

'Develop an attractive and functional web application' is the foremost part of this aim, the main platform in which users will interact with my product is the web app. If it is not both attractive and functional users may have little or no interest in using it again.

The attractiveness of a Web Site is highly subjective, different people will have different opinions on the aesthetic qualities of a page. Therefore it is difficult to define the success criteria for this aim. I believe the user testing gives clear evidence that my web site design is not only attractive but also usable to most DOTA 2 players.

Drafting Assistant

The next part of this aim is to develop a 'reliable and helpful drafting assistant'. The reliability and usefulness of my drafting assistant comes down to the data it uses and how it processes this data into accurate results. The database currently holds the results of nearly 100,000 games, which is a rather large sample size in order to pull hero statistics. The games processed from the API are also from the latest version of DOTA 2 and therefore are representative of the current state of the game. The processing of the data uses win rates and the mean averages of those win rates to calculate advised pick and ban decisions.

XML Parser

In order to 'import data from Valve's API' I was required to develop some way of parsing the XML data from a game into the data layer. Originally I made the decision to learn and implement a DOM parser as I had started work on this concept in my Web Technologies module[19]. I taught myself aspects and implemented a parser to retrieve data on heroes beating heroes and then insert the correct data into my MySQL Database.

As the module progressed we started to learn SAX, an alternative XML parsing technique, and I realised that implementing a SAX parser would parse games much more efficiently. Currently whilst my XML parser is functional and has parsed to this date close to 100,000 games, if I had implemented a SAX parser it would improve the efficiency and performance of the system.

I believe I have achieved the goal of being able to 'import data from Valve's API' although if I had more time I would implement a SAX parser to improve the performance of the system by allowing it to parse games at a quicker rate. Although this increased speed would only be valuable when adding large amounts of past games, as the currently implemented DOM parser can parse games much more quickly than they occur. I conclude that keeping up to date by parsing games as they finish is easily achievable with the DOM parser.

With more time I would develop a full SAX parser for each element of the database in order to parse a large amount in a shorter space of time. This would increase the efficiency of the system when filling tables with large amounts of previously occurring games.

A connected Android Application that extends the websites functions and allows the user to use the drafting assistant on a mobile device

I have implemented the statistics pages on the android platform in order for them to be used by a wider part of the client base. This has allowed me to develop my ability to implement

systems on new platforms. It has also met the requirement of extending my web application onto a mobile device. I do feel, however, that the android application would be much easier to implement on a larger screen, possibly a tablet, due to the amount of user input required.

Given more time I would extend my android applications amount of functionality, implementing my drafting assistant and game onto it. I would also improve the design and aesthetics of my application to make it simpler and more enjoyable to use. In my opinion, this would improve the popularity of my application and its usability. I would also design the system for use on a larger device, as I feel it would be both easier to use and more natural feeling on a tablet sized device.

6.1.2 - Recommended Aims

A personalised log in system that allows the user to select certain heroes that they like to play and heroes you do not like to play, so that they are less likely to be suggested in a draft

After careful consideration I decided that a personalised log in system would not add the extent of useful functionality I originally believed. The system would probably feel fairly clunky, it would be difficult to implement a system that advised certain heroes more frequently than others. For example if a hero was a poor decision in a certain situation, but it was one of the users 'preferred picks' could the system apply a multiplier to the hero and inform the user to make a poor decision.

I came to the decision not to implement the log in system due to the work required to implement the system and the lack of functionality and usefulness it would bring.

A page that shows the user their recent games, and analyses how they draft including patterns in their bans/picks and suggests ways to improve your drafts with heroes they should ban/pick more

I found that collecting and storing a particular user's games to be more technically complex than I originally foresaw. The system would have to parse through every game that was played and store the user IDs of every player in that game. This would vastly increase the time the parser takes to parse a game and require vast amounts of storage in the data layer, assuming I would store data for each user found.

I decided against implementing this due to the technical complexity of the functionality being vast in comparison to the value it would add to the system.

6.1.3 - Optional Aims

A 'game' inside the android app that allows a user to draft against an AI and try and out draft them, victory will depend on statistically whether the user picked correct heroes

The above aim began as an optional aim and developed into one of the main components of the system. This was down to a change in my opinion on the value of this component. I believe the game is both highly engaging and helpful to a DOTA 2 player who is looking to develop their drafting skills.

As DOTA 2 games can take any amount of time between twenty minutes and two hours players usually take small breaks between games in order to wait for a new game to be ready. As a result, small engaging DOTA 2 related mini games have proven popular with the community. Due to this I believe my game would be incredibly useful and popular with the intended user.

My game is engaging and useful due to the speedy nature in which a user can play and receive a result. There are also multiple difficulties which will enable a user to improve over time by attempting to beat each of the difficulties. The difficulties are well defined and as shown in my testing are a good representation of three tiers of knowledge and ability allowing any level of player to learn and develop from the game.

With more time to work on the drafting game I would improve the difficulties including the ways in which they differently counter picked the user, mainly at the more difficult end. A more intelligent difficulty that chose based on and increased number of factors alongside more expected factors would improve the usability of the system for the highest level of player.

Advanced statistical pages on your games, hero trends, the standard deviation of hero win rates ect.

As I have previously mentioned storing data for one individual user is highly complex and technically demanding. Due to this I decided to alter the aim to not be specific to each individual user, but specific to each individual hero. My advanced statistical pages show the standard deviation of an individual heroes win rates over time, win rates against individual heroes, as well as the best and worst hero matchups for that hero.

I believe these pages add valuable information to someone attempting to learn about the game. The pages enable the user to analyse which heroes are improving and which are

getting worse as well as keep up with the state of the 'meta game', the current trends that impact hero selection.

6.2 - Impact

6.2.1 - Social

The social impact of the project is, in my opinion, vast. The system has the potential to be highly useful and advantageous to over 10 million users[20]. It could potentially improve the ability of a large number of DOTA 2 players and therefore contribute to the ever growing professional scene. There is currently a lack of tools to help players improve at DOTA 2 drafting, an element which is crucial to the game, hence why I believe my product will have a large impact on the community.

The potential impact and popularity inside the community is highly dependant on the stretch of my product. Posting the product onto popular community forums and sites, such as reddit[21] and dotabuff[22], could attract the attention of largely respected community members. If professional players find the site useful it could lead to their fan base using the site and result in exponential growth of the sites client base.

6.2.2 - Commercial

The commercial impact of my product is largely dependant on the usage. Due to my opinion for the demand of a product like mine and the potential client base, if supported my product could be a popular tool for upwards of ten million people[20]. If the support and resources were given to my product in order to support this kind of use, advertising revenue could be a generated.

Advertising in the video game industry is huge, with large international video game related companies investing in significant web-based advertising campaigns. For example the video streaming service Twitch, predominantly used for the streaming of live video gaming, was bought by Amazon last year for just under one billion dollars[23]. This shows the scale of the online video gaming related advertising market, because the majority of Twitch's revenue is from the advertisements it shows to users before and during the live games they watch.

6.2.3 - Economic

The economic impact of the project is very similar to its commercial impact. The product has the potential to be profitable whilst fulfilling a demand of a large emerging market. Although some of the technologies I have used are not necessarily relevant at present in the technology economy. For example JSP is not a commonly used technology in the real world due to its high technicality and hosting support. However Java, MySQL and Android are widely used technologies with high levels of support.

6.3 - Complexity

The system has been developed using a significant amount of complex technologies, some of which I have learnt during the third year of my degree course and others I have taught myself.

The use of an apache tomcat server and JSP's has been challenging as I had never worked with servlets until I began the project, my understanding of servlets continued whilst I was studying Web Technologies. The development of an XML parser was a similar story, I had to teach myself parts of the DOM technology until I continued my learning during my Web Technologies module.

The development of a game that makes decisions based on user input alongside complex decisions which change based on many factors, is something I was unexperienced in prior to beginning the project. Developing the game component of the application has developed my abilities in both Java and AI based systems.

The android application has added further complexity to the system and required me to research and understand new platforms. Developing in the android environment has taught me valuable skills in an emerging technology that I intend to implement in my ongoing professional development.

The advanced statistics involved in my statistical pages were a subject I had to relearn and research. I learnt a large majority of those statistics during my A-level studies but had to revisit the usefulness of certain metrics on varying data.

6.4 - Personal Development

Developing this system has helped me to enhance my skills in a number of areas as well as enabling me to learn new valuable skills which will be of use during my imminent career. Being afforded the responsibility to plan, design and implement a complex system has improved my confidence in software development, an area in which I previously lacked experience.

My ability to plan and document requirements for a system has been developed immensely throughout this project. I have recognised areas and aspects that I planned well and those which were planned poorly. The aims and objectives for the project changed significantly throughout, however my plan has been detailed enough in order to accommodate for changing goals. If I were to plan the project again, I would ensure an increased focus on functional requirements in order to have a clearer picture of the functional implementation of the system.

Throughout the project I believe the skill that I have been developed to the greatest extent is my programming ability. This is mainly due to the project being the first time I have had sole responsibility for the development of a complex system. My confidence and skills in technologies such as Java, html, JSP, MySQL, Android and XML have significantly improved. I have also developed my abilities in research and self development as I have had to learn and develop my skills independently. I feel this independence has prepared me for a career in the technology industry as an understanding of a number of relevant technologies is vital.

I have also developed the valuable experience of software development, from planning to implementation, and the issues that can occur during every aspect of such development. This knowledge, I am certain, will be invaluable throughout my post education career.

7 - Conclusion

I believe the system has met all of its essential aims and a number of its recommended and optional aims. During implementation it became clear that achieving all of the optional aims was not viable in the timescale given. Therefore I decided upon those aims that would provide the most useful functionality to the system. For example I chose to exclude the twitter/reddit news page as it does not contribute to the main objective of the system. Whereas the Drafting Game directly links to the systems main objective, the drafting assistant, by allowing users to practice against an AI.

Given more time to further develop the system I would like to improve the functionality of the android application as it does not contain a large amount of the functionality included in the web based application. I would also like to develop the way the system collects data, as I believe that by implementing and running a SAX parser, the system would be more efficient in keeping the data relevant.

I also believe that the system has met all of its non functional requirements. The system is highly usable due to the aesthetics and web design of the system being altered following relevant user testing. It has also proven to be highly portable due to the fact it has been ported to a second platform, android. The system is also highly reliable due to the technologies it uses, for example, MySQL has built in safety measures to prevent data loss and corruption.

Further development would enable me to improve system testability by defining a number of further complex tests which would confirm each of my methods gathered and manipulated data correctly. I would also improve the modifiability of a certain methods, as some of my methods rely on other methods remaining the same.

Developing this project has been both fun and rewarding. I have enjoyed both the planning and programming stages of the process. It has given me a great amount of experience in software development and researching and learning new technologies. In conclusion I believe the system is a great success and will have large appeal to its target audience, having met the functional and non-functional requirements.

8 - Bibliography

[1] http://en.wikipedia.org/wiki/Dota_2 - DOTA 2 Article, Wikipedia, May 2015

[2] http://www.valvesoftware.com/ - Valve Corperations Website, Valve Corporation, May 2015

[3] http://dota2.gamepedia.com/Heroes - Heroes Article, Dota 2 Wiki, May 2015

[4] http://dota2.gamepedia.com/Game_modes?cookieSetup=true - Game Modes Articule, Dota 2 Wiki, May 2015

[5] http://dev.dota2.com/showthread.php?t=47115 - Dota 2 Match History WebAPI, ZoID (Valve Developer), Sep 2011

[6] http://en.wikipedia.org/wiki/Functional_requirement - Functional Requirement, Wikipedia, May 2015

[7] http://en.wikipedia.org/wiki/Non-functional_requirement - Non-Functional Requirement, Wikipedia, May 2015

[8] http://en.wikipedia.org/wiki/Multitier_architecture - Multitier Architecture Article, Wikipedia, May 2015

[9] https://www.novell.com/documentation/nw65/web_mysql_nw/data/aj5bj52.html - Benefits of MySQL, Micro Focus, November 2009

[10] https://www.mysql.com/why-mysql/presentations/mysql-replication-reliability-self-healing-crash-recovery/ - MySQL replication, reliability, self healing and crash recover, Oracle corporation and/or its affiliates, 2015

[11] http://www.dotabuff.com/matches - Matches of DOTA, Dotabuff, May 2015

[12] https://dev.mysql.com/doc/refman/5.0/en/integer-types.html - Size of integers in MySQL,Oracle Corporation and/or its affiliates, 2015

[13] https://dev.mysql.com/doc/refman/5.1/en/table-size-limit.html - Size of MySQL Databases,Oracle Corporation and/or its affiliate, 2015

[14] http://bielik.blogspot.co.uk/2011/05/seven-advantages-of-java.html - Seven Advantages of Java, Miloš Bielik , March 2011

[15] http://Javaontips.com/10-benefits-advantages-of-jsp/ -10 Advantages of JSP, VINEET TALWAR, September 2011

[16] http://getbootstrap.com/ - Bootstrap, @mdo & @fat, May 2015

[17] https://docs.oracle.com/Javase/7/docs/api/Java/util/Arrays.html - Arrays, Oracle Corporation and/or its affiliates, 2015

[18] http://www.dotabuff.com/heroes/winning?date=patch_6.84 - Hero win rates, Dotabuff, May 2015

[19] https://campus.cs.le.ac.uk/teaching/resources/CO3098/ - Web Technologies, Stuart Kerrigan, January 2015

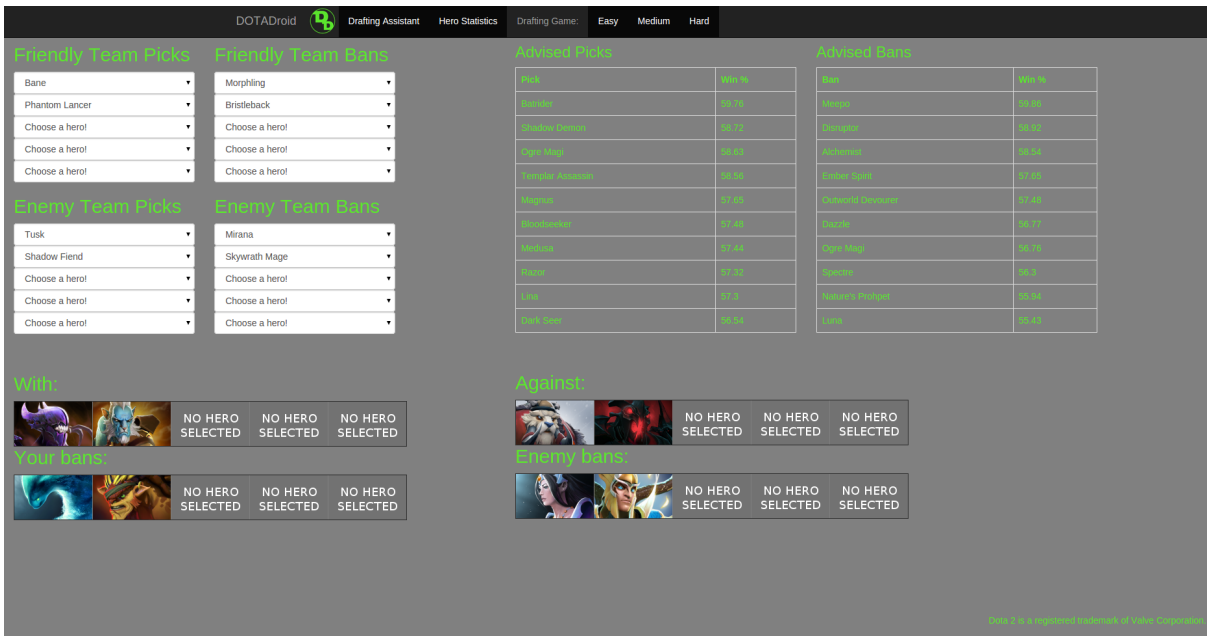
[20] http://www.theverge.com/2015/1/17/7628755/dota-2-now-has-over-10-million-players - Dota 2 now has over 10 million players, T.C. Sottek, January 2015

[21] http://dotabuff.com - Dotabuff, Dotabuff, May 2015

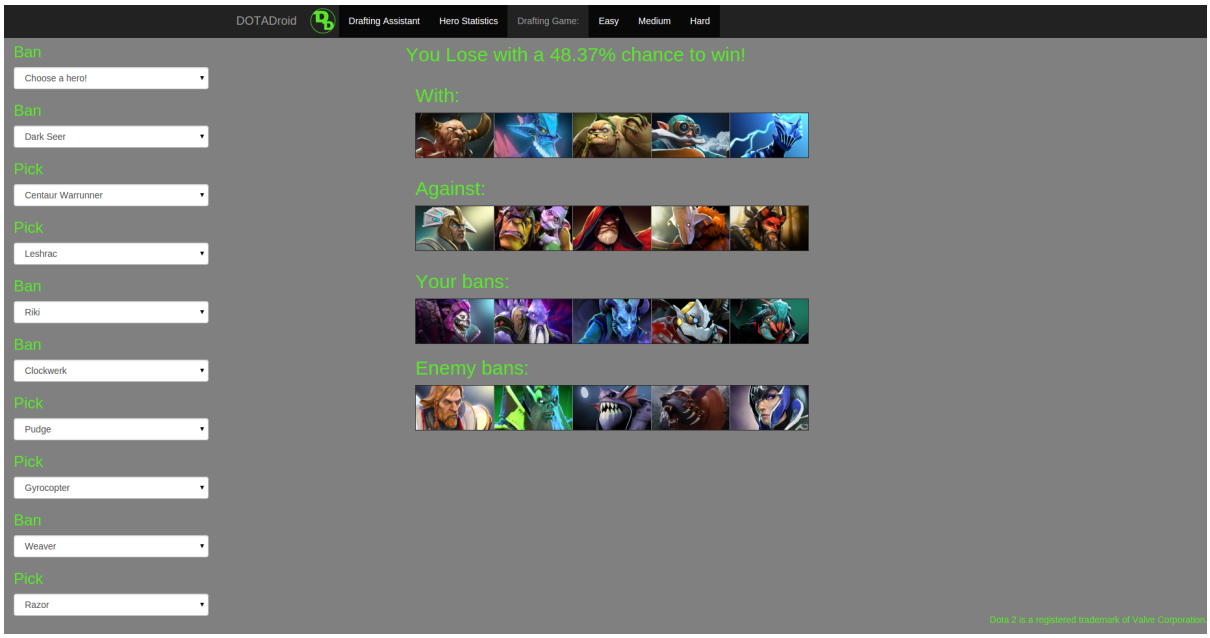
[22] http://reddit.com/r/dota2 - DOTA 2 Reddit, Reddit, May 2015

[23] http://www.bbc.co.uk/news/technology-28930781 - Amazon buys vIDeo-game streaming site Twitch, Kim Gittleson, August 2014

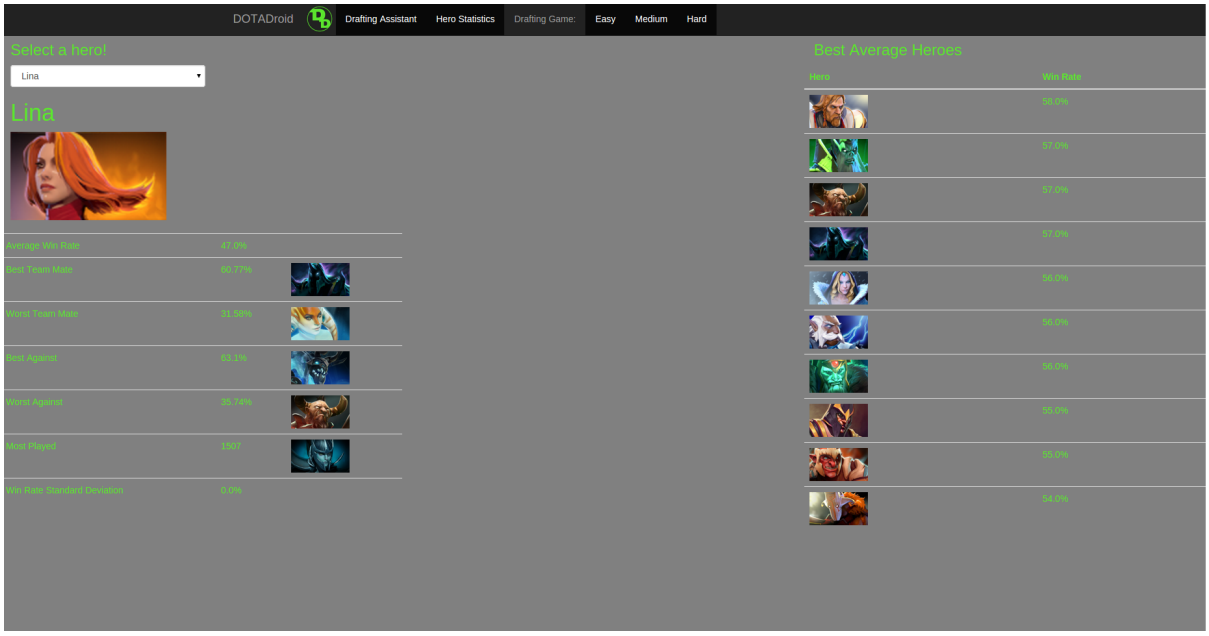
9 - Appendix



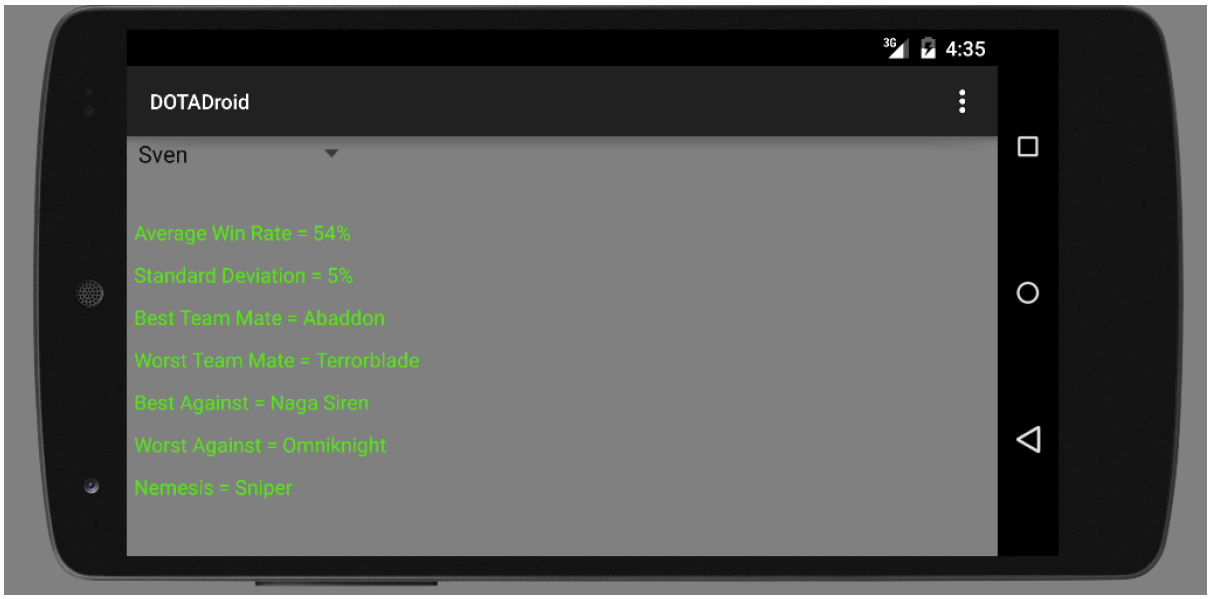
A screenshot taken of the drafting assistant



A screenshot taken of the drafting game



A screenshot taken of my statistics page



A screenshot taken of my android application

9.1 - List of Figures