Football Match Prediction Web Application - SureThing

Marina Shchukina

A report submitted as part of the requirements
for the degree of Computer Science
at the School of Computing Science and Digital Media
Robert Gordon University, Aberdeen, Scotland

February 2015

Supervisor Dr. Robert McDermott, Dr. Richard Glassey

Abstract

The aim of the project is to build a web application simulating the football betting experience and addressing two main issues. Firstly, filling an existing void for a system that makes football match prediction customizable and transparent to the user. For each upcoming match, the application will provide the user with an overall probability of either of the two teams winning the match expressed in percentage. The user would then be able to influence this probability value by adjusting the weights of the factors that contributed to that result. Depending on the values of the weights, prediction may be different for different users. In the long term, users of the web application would be able to create their own "betting system" by constantly re-adjusting the default weights and hopefully coming up with a set of weights that give the best results. Secondly, allowing the users to analyse their past performance and compare their results and prediction weights with the other users of the application.

The stated above would be achieved by taking several steps. First, the background research will be carried out. On completion of the research, current similar websites will be researched and a set of requirements will be created to assess users needs. After that a layout and overall design of the application will be produced, as well as the desired behavior of its features. Once the prototyping is completed, the main project deliverable, i.e. working web application will be developed and throughly tested.

Acknowledgements

I would like to acknowledge and extent my gratitude to the following people who have made the completion of this project possible:

Dr. Roger McDermott for his support and guidance in this project Dr. Richard Glassey for his initial help and valueable advice My husband Murray and baby daughter Scarlett for their support and patience.

Declaration

I confirm that the work contained in this BSc (Hons) project report has been composed solely by myself and has not been accepted in any previous application for a degree. All sources of information have been specifically acknowledged and all verbatim extracts are distinguished by quotation marks.

Signed ..... Date .....

Marina Shchukina

Contents

List of Tables

List of Figures

Contents

Chapter 1

Introduction

A short paragraph introducing the topic the chapter examines.

1.1 Background

A number of pages about the background of the project.

1.2 About this Thesis

This is the thesis of *Insert Full Name Here*, submitted as part of the requirements for the degree of MSc Computing: Software Technology at the School of Computing, Robert Gordon University, Scotland.

A number of paragraphs detailing the main expectations of this body of work.

1.3 Conclusion

A short conclusion summarising the chapter.

Chapter 2

Background studies and objectives

In this chapter the background of the project will be discussed. The chapter will take its reader through the history of sports betting with a particular focus on online gambling. The primary and the extended objectives of the project will be outlined and the professional considerations will be addressed.

For this chapter put lots of references (especially in the BG subchapter)

2.1 History And General Information

Hello!!

2.2 Objectives

People have always been interested in games with the element of luck and therefore gambling is one of the oldest forms of entertainment of mankind. The rise of the Internet and mobile devices has made remote gambling more available for a wide variety of users. The reason for that could be that Internet applications and websites can be are easily accessed 24/7. Amongst the most popular types of online gambling can be found card games, dice games, electronic games (such as poker), betting on sporting events, etc. Sports betting is no longer associated solely with horse racing. Among all types of sports gambling, football gambling is a leading industry with a share about 70When it comes to any sports betting (football betting including), the user is trying to predict the result of the event and placing the money on the outcome. This prediction can be made based on a hunch or by using logic and domain knowledge, in a lot of cases by both combined. Naturally, this gave rise to a vari- ety of betting software systems that are attempting to predict the next match re- sult. Most of the time those betting systems work as a black box not allowing the user to influence the prediction output and preventing the user from understand- ing the exact logic used inside the system. Football bettor can have various strategies when making a betting decision. As mentioned above, the user can buy a prediction software and simply follow the tips suggested by that system. Another option is to make a decision influenced by the opinion of the other tipsters, experts opinions or rumours. However, if the aim is to achieve sustainable profit (or minimise the loss from betting) , most experi- enced bettors

would ignore betting tips and predictions of others and go for the pure facts trying to make their own prediction. To make this happen, the bettor has to aggregate several pieces of information from various sources. This action has to be repeated for every single match. From my experience, the necessity to repeat an action many times could lead to a creation of interesting software solutions. That is how I got inspired to create an application that would aggregate this information for the user and therefore act as an interactive decision supporting system.

2.2.1 Primary Objectives

The purpose of my final year project is to create a web application that can help users to predict football match results and make profitable bets. There are many similar websites and applications. However, I think my application is different from the other ones. The key feature of the app is the fact that the prediction output is transparent to the user and can be easily adjusted and customised.

2.2.2 Extended Objectives

Hello!!

The purpose of my final year project is to create a web application that can help users to predict football match results and make profitable bets. There are many similar websites and applications. However, I think my application is different from the other ones. The key feature of the app is the fact that the prediction output is transparent to the user and can be easily adjusted and customised.

2.3 Problem Specification

2.3.1 Limitations and Evaluations

Hello!!

2.4 Social, legal and ethical issues

Legal Finding free API for a newly created betting application may become a non-trivial task. Therefore, I am considering using web scraping as one of the options to load most recent football data into the system. It can be said that there is a fine line between collecting information using the web scraping technique and stealing it. Most of the websites have a copyright disclosure defining the rules for the use of the information they provide. Thus, I will carefully read the disclosure statements and follow them along legally and ethically.

Ethical Due to the nature of the application it is inevitable that it will store some basic user data in its database. The application must take all the necessary precautions to protect the stored data and sensitive information. The application will not disclose personal data of its users to any third parties.

Social There are several advantages of using the application for a rational bettor. First, the use of it will hopefully lead to more profitable football betting and also reduce the amount of thoughtless

bets. Secondly, the use of application will save time spent on gathering information before making a betting decision.

Professional Although the main aim of the application is to provide transparent prediction to the user, there is still certain amount of calculation happening in the background. I assume that user will trust the betting system when making a betting decision. Therefore ensuring the accuracy of the calculations and providing good test coverage is a very important part of the application development.

Hello!! a wee summary: what we discussed in this chapter In this chapter we discussed the history. I also introduced the objectives of this paper and lined up several use cases making clear for the user how the application is going to work.

In general, the application is not aimed to promote gambling. Moreover, it supports a more sensible and measured approach to football betting. In this chapter we discussed(the main conclusion for the chapter)

Chapter 3

Requirement Analysis

Before designing a piece of software, it is important to hold good understanding of the project requirements. In this chapter a number of functional and non-functional requirements will be listed covering all the aspects of the future project.

3.1 Target Audience

The intended users of the application developed will be people with interest in football. More specifically, people interested in predictions of football results and football betting. With this in mind, the age range of potential users will be 18 and up.

3.2 Target Audience Questionnaire

The target audience research aims to gather information on how... Due to the spread of target users, an electronic questionnaire was used to collect the results. A full break down of the questions asked and the answers received can be found in the appendices, reference X.

3.3 Researching the Potential competitors

Before gathering the project requirements, it is good practice to conduct research on what current websites are already available to football fans. The research can be a source of inspiration and would also help to avoid possible design mistakes. During the analysis, it is important to attempt to understand the main purpose of the analysed applications, as well as the way they present information and communicate with their users.

This section is concerned with websites that can be useful for predicting football results. In our context, these are the online sources of information a football punter would turn to before making a betting decision unless the decision is based solely on punter's intuition. In general, several different types of such websites can be found online, namely:

1. Sports news websites

2. Football statistics websites
3. Websites of betting providers
4. Communities
5. "Black-box" prediction applications

This section of the report looks at one or two examples of each of the categories presented above, analysing the weak and strong points of the chosen website and discussing usefulness of the whole category from the point of view of a footaball punter.

3.3.1 Sports news websites

This category represents football news websites. Into this category belong both general sports websites having a football section and football news websites, for example:

- BBC Sport - http://www.bbc.co.uk/sport
- The Guardian Sport News - http://www.theguardian.com/uk/sport
- Sky Sports - http://www.skysports.com/
- The Times Sport section - http://www.thetimes.co.uk
- Football365.com - http://www.football365.com
- Goal.com - http://www.goal.com

The sports news websites aim to present the news alongside the essential football statistics. The information is usually not as detailed as on the football stats websites, however the very latest football news compensates for this drawback. Each of the *news* websites named above (BBC News, The Guardian, Sky Sports) have a football section that presents the reader with the combination of football news and stats. One of the most popular football news websites is BBC Sport Football.

3.3.1.1 BBC Sport Football

BBC Sport Football is a brilliant news website. It offers very neat and simple interface and does not overwhelm the reader with irrelevant graphics. Although, it almost looks too minimalistic, the user can still find the most important information about football teams and players. The website provides authomatically updated live scores accross all featured football leagues.

BBC Sport has a very impressive news coverage of both major and minor British football leagues, as well as the main european leagues. It can also take pride in high quality writers (journalists) contributing to the website. An interesting feature is videos embedded in the webpage.

As already mentioned above, the main drawback is that the stats are kept to a bear minimum. This can be a problem for a passionate football fan that want to analyse the details of the game from all possible angles. However, for a purposes of a punter this level of statistics should be sufficient.

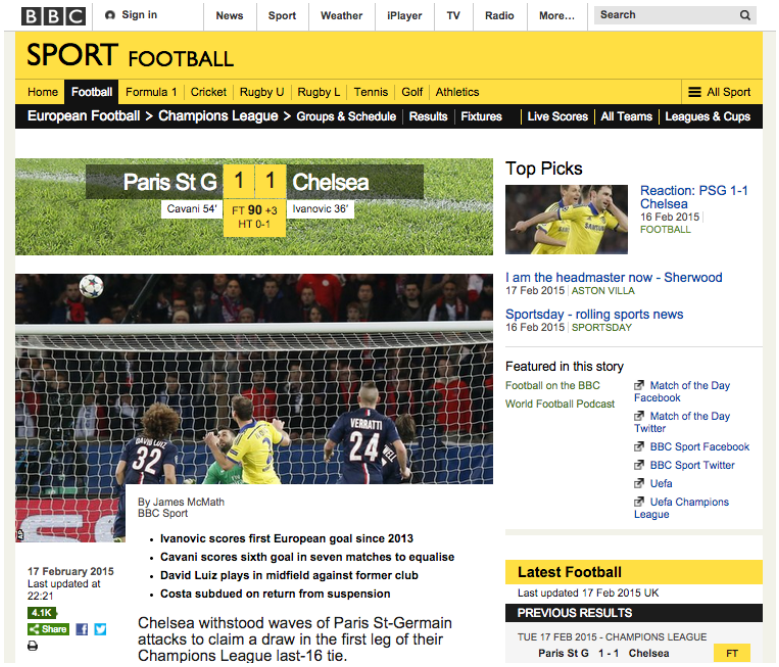

Figure 3.1: BBC Sport - Football section

3.3.2 Football statistics websites

Football statistics websites focus on detailed statistics and analysis on football matches, teams and players. These are some examples of such websites:

- WhoScored - http://www.whoscored.com
- Squawka - http://www.squawka.com
- WhoIsInjured - http://whoisinjured.com

3.3.2.1 WhoScored

Among all football stats websites I have analysed, WhoScored is one of the most impressive ones. It has a lot of statistics, but most of it seems to be quite relevant. The website is extremely well designed, and its navigation is intuitive. The website is offering statistics and deep analysis on the major European divisions, as well as providing data on over 500 leagues and 15,000 teams. As to the data source, the website is supported by Opta, the biggest and a very reliable live sports data company standing behind BBC Sport, Sky Sports, etc.

The way "WhoScored" presents information on particular matches to the user can be very inspirational for this project.

3.3.2.2 Squawka

Squawka is another websites worth looking at. It is an award-winning application for football fans that uses real-time data visualisations to explain the game. The main idea behind is to help users to understand the game.

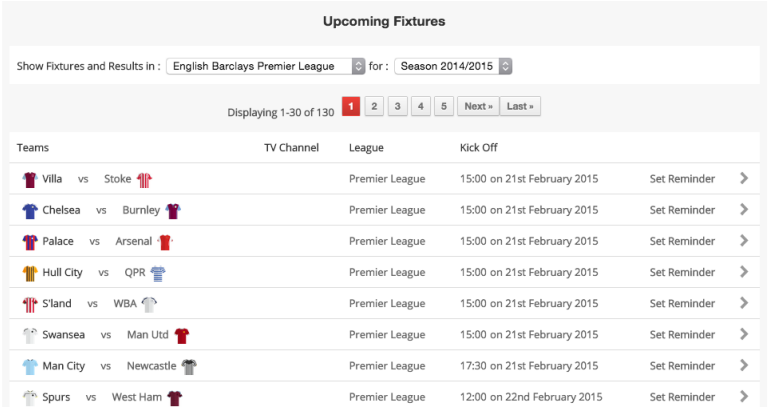

Figure 3.2: Squawka

From a visual point of view, Squawka has a nicely designed, pleasant interface. However, it is a little bit heavy on the client-side (Javascript), which contributes to sometimes slow performance. Another downside of the website is an extensive amount of adverts that distracts the user form the main content.

3.3.3 Websites of betting providers

In this category can be named many websites.

3.3.4 Communities

Communities is an category of the websites with an interesting idea behind. These websites are specialised social networks for sports fans and punters. There are two websites in this category:

- OLGB Betting Community - http://www.olgb.com
- Vital Football News and Fans community - http://www.vitalfootball.co.uk

3.3.5 Black-box prediction applications

http://www.predictz.com/predictions/europe/champions-league/609828/ http://www.vitibet.com http://www.forebet.com/en/ small prediction tool at goal.com: http://www.goal.com/en-gb/match/psg-vs-chelsea/1974582?ICID=FX

Firstly, there are websites containing detailed football statistics and its analysis. Some of them may also attempt (not nessesarily) to make predictions based on these statistics, however, this is not the main purpose of such websites. among

After having analysed the above websites from various categories, I came to the following conclusion. A well-chosen combination of several websites will be definitely able to provide enough information to make a thoughtful betting decision.

Most punters would have their own football betting system (betting strategy). Although even the best system cannot guarantee success, it can greatly increase the prospects of making a profitable bet.

Therefore, when predicting football match result, a potential punter will have to make a small research each time. The aim of such research would be to make note of the relevant statistics for the teams involved in the game (depending on the "input variables" of the betting system), for example previous results for both teams, their performance at home and away, players statistics (injuries, suspensions, transfers), recent change of team managers, etc. The problem is that many football stats websites offer way too detailed information for this purpose, therefore the relevant stats needs to be "hand-picked" from several sources for each match.

The application will attept to put all the relevant statistics in one place and break the information down into logical "prediction modules". In addition, the application will enable users to pick the input variables and assign them a weight of user's choice, representing the importance of the input variable for the match results prediction.

3.4 Requirement Analysis

Computer Systems life cycle consists of five phases: Requirements and Analysis, Design, Implementation, Testing and Evolution. Then, after Evolution, the cycle can start again with an additional set of requirements. Therefore, the Requirements analysis is a crucial part of the project and the cornerstone of the software development life cycle. Requirements analysis involves gathering information in order to meet customer needs and defining what the future application is expected to do. This phase is especially important in the real world environment, when developing an application for a customer. In this case, clarifying the requirements in the early stages of the project would help to ensure that both sides understand and agree on the feature set of the future application.

Although it is not very likely that requirements for this project will change during the development process, defining requirements can be very beneficial. The detailed requirements analysis will aid understanding how different parts of the project are expected to interact with each other, as well as how the application will communicate with its users.

For better transparency project requirements have been split into functional and non-functional requirements.

Before outlining project requirements, I would like to start with some definitions relevant to the whole project.

3.4.1 Definitions

Application Football League - in order to reduce unnecessary complexity at this stage of the project, the application will be only supporting one league (Premier Barclays League)

Matches Overview a list of upcoming and played matches presented on the main page of the application

Dashboard - an interface available to authorised users, containing a list of played ("archived") and upcoming matches saved by the user

Prediction Module my own term, standing for blocks of football statistics for each of the playing teams (for example, league position), that are evaluated against each other and based on the result of such evaluation, a match prediction (in favour of one of the teams) is made. Application calculates "prediction value" for each of the modules.

Match Result Prediction - to calculate a match result prediction, each prediction value is first multiplied by its weight. The weights would be different for each user unless default system settings are being used). Then, all of the weighted values are added together and based on this value a prediction is made.

Default System Settings - application has a set of default weights for each prediction module. These weights are used in the prediction calculation by default.

Default User Settings - each user of the application can save a set of prediction weights that are different from the default system settings. From the moment these weight are saved, they will be used by default for each match committed by the user.

Match Specific Prediction Settings - each user of the application can also save a set of prediction settings specific for only one match.

Match result - "Home Win" in case of the win of the hometeam, "Away Win" in case of the win of the awayteam, "Draw"for the draw

3.4.2 Functional Requirements

Functional requirements describe the behaviour of the application in terms of its functionality. These are the "must have" functions of the application addressing the business targets that application must satisfy. Good functional requirements must be concise, complete and unambiguous.

In order to add structure to the design and development process, the project was broken down into high level features of the future application. The functional requirements are grouped by the functionality related to those features.

3.4.2.1 General Web Application Requirements

These are the requirements related to the basic functionality of the web application, such as account registration, login and account management.

- The application should allow users to register and create a new account with the application
- User will be able to register using a standard web form
- For the registration purposes user will provide a valid email address and a password
- User will confirm a password in a separate input field
- After a successful registration application will send an email containing a confirmation link to the user
- On a successful confirmation of an email address, user will be successfully registered

- In case of any technical problems with the initial confirmation email, the application will be able to issue a new email and send it to users on their request
- The application should allow users to sign into their accounts using a standard web form
- When signing in, user should provide valid credentials, otherwise an application will throw a validation error
- The application will allow users to manage their account by changing personal information (for example, location, favourite football team)
- The application will allow users to change the email address associated with their account
- The application will provide an option to change user password for security reasons

3.4.2.2 Matches Overview

Below are the requirements relating to the matches overview.

- On the main page of the application user will be presented with a list of upcoming matches for the current season in the league
- User will be also able to view a list of played matches and switch between upcoming and played matches using navigation tabs
- For each of the unplayed matches user should be able to navigate to the match page and see more details about the match
- From the main page user will be able to save any unplayed match to the dashboard
- For each of the played matches user will be able to navigate to the match page and see more details about the played match

3.4.2.3 Prediction

The predicted outcome of a football match in the application will be calculated after evaluating several factors that can influence the match result. An example of such a factor could be a previous match result, league position of each team or a team's performance at home. Each of those factors is represented by an equivalent "prediction module" in the match view. These are the requirements related to the prediction of the overall match result.

- To calculate prediction values for each module, application uses default system settings in absense of default user prediction settings or match specific settings
- Application uses default user prediction settings in absence of match specific settings
- Application uses match specific settings in case user set them for this match

3.4.2.4 Upcoming Match View

Users should be able to view detailed statistics of the upcoming match. These are the requirements related to this functionality.

- The information presented in the unplayed match view will be divided into two major sections: prediction modules and additional information
- For an unplayed match user should be able to see a set of prediction modules
- Each module should contain relevant piece of football statistics
- For each module it should be clear which team is a winner for the module and what is a probability for this outcome if the prediction wuld be based solely on this prediction module re will be several prediction modules and the football information for each team The details will include statistic data for each of the playing teams, such as team standings, results of 6 previous matches, results of previous 6 matches played at home, results of previous 6 matches played away, results of previous meetings with the opponnent within current season.
- from the prediction module user can see stats of the website population

3.4.2.5 Played Match View

After the game has been played, the application will offer prediction statistics and some feedback to the user. Hopefully, this will allow users to compare their betting strategy with the fellow punters and analyse their own betting settings. These are the requirements related to the played match view.

-

3.4.2.6 Dashboard

Dashboard is a key view of the application.

The idea behind the dashboard in this application is similar to online shoppping experience: user saves an item to the shopping basket and from the basket can submit a purchase or cancel it. In our case, user browses through the matches overview and save matches to the dashboard for a latter review. The requirements outlined below describe the dashboard functionality in more detail.

- from the overview of the matches on the main page, user can add a match and save it to the dashboard
- user can delete matches from the dashboard
- user can override the system's recommended prediction modules weights (called "betting system settings") using the" betting system" tab in the dashboard. Any next match saved to the user dashboard will be shown with these user default betting settings.
- user can override the user default betting settings and have different betting settings for each saved match. User can save a match with the new weighting percentage.

- After completing the betting settings, user can commit to bet a saved match. After that match betting settings cannot be changed.
- If the match is saved in the user dashboard and it changes its status from "unplayed" to "played", application will evaluate user's commit and estimate whether user won a virtual bet or not based on the actual match result.

User can commit to bet a match Committed games are marked in the dashboard (red dot) Played games are marked in the dashboard (grey font) Once user committed games has started (observer, onchange on a filed), it should indicate the winnings and write to db Fix the kits

3.4.3 Non-functional Requirements

Non-functional requirements specify how the system is going to perform.

3.4.3.1 Testability

The application should be testable the behaviour of the application should be verifiable through a suite of automated tests. A set of test cases will be written for the application, testing features such as logging in, account registration, integration with third-parties API used in the application, as well as various scenarios of database models interactions. In addition to these tests, the business logic in the match prediction must also be predictable, consistent and testable.

Maintainability Scalability Documentation Performance Responsiveness Usability Usability means how easily can users learn how to use an application. The key would be to reduce an effort to understand

3.4.3.2 Security

The nature of the application requires from user to provide some confidential information. Users need to be assured that the confidential details remains secure, and that the system is protected against XSS, CSRF and SQL injection attacks.

graceful degradation? accessibility security

3.5 Overall Architecture

Provide a list of all the chapters within the thesis and a brief summary of the content.

Design of an application a a whole, overall design (just boxes and lines) **A**rchitectural diagram (overview) (aosabook.org/en/moodle.html -example), quite high level

3.6 Choice of Third Parties API

After analysing the functional requirements, it became apparent that this type of application will need the latest football data in order to operate correctly. The easiest way to load such data into the

application would be to integrate the application with a third party API. The process of finding an appropriate API for the project will be described in this section.

After brief research, one thing became apparent. Live football data is a very desirable product and therefore it is not easy to find free of charge live football data API. The key problem is that the data the application needs has to be very recent. Real-time data in particular is very expensive, because of its use by the gambling industry for betting on various markets as the games are going on. It is much easier to find free historical football data.

This is a (not complete) list of API providers that I researched about.

- Optasports.com (http://www.optasports.com) Opta is an industry leader. It provides a wide range of XML feeds covering many sports. The feeds include fixtures and results, live scores, live player stats and many more. Data provided by Opta is very reliable and is used by top-notch clients, such as BBC Sports, BT Sport, Sky Sports, as well as many betting providers and newspapers.
- Openfooty (http://www.footytube.com/openfooty/service.php) Openfooty is an interesting project with very detailed API documentation. However, a quick look at the developer forums shows a stale community and many questions about why no one seems to actually be able to get a developer key. Unfortunately, I also did not manage to obtain a key for this API.
- football-api (http://www.football-api.com) is a paid API service but does offer the English Premier League endpoints for free (demo use). The API will restrict by IP addresses and limit calls based on your package. Includes endponts for Competitions, teams, standings, live scores, fixtures and commentaries. See the pricing page.

However, there can be found many high quality APIs that are not for free. how I came to choose the API I am using in the application

3.7 Project Plan

The project progress timetable is presented in the Gantt diagram below. I found it appropriate to set In general, my two main milestones will be completing the first prototype of the application before the Christmas break and completing the second prototype by the end of April, 2015 (this includes all the testing and bug fixes).

The first prototype will have implemented most of the basic features of the application (my development part is broken down into features - viz. the Gantt diagram). The second prototype is the final version of the application; it will include all the planned features and the graphical design. I will try to make an even progress on the report throughout the whole time available, as it can be seen from the diagram.

3.8 Conclusion

A short conclusion summarising the chapter.

Chapter 4

Application Prototype

Before making a start on of the implementation phase, a lot of effort was put into the creation of the application prototype. Prototyping is a process of developing the initial model of the future application in order to determine the correct application structure, its functionality and the general concept. A prototype is just a model and may differ from the final product.

The project requirements outlined in the previous section of this report were used in order to create a mind map representing the navigation between the pages. This helped understanding what exactly is expected to be seen on each page of the website and what is the user journey in terms of the navigation. Wireframes were created for all the pages of the website. For this purpose was used just paper and pencils to aim flexibility. Agile was chosen as the methodology for this project. In general, Agile methodology focuses on team communication and project transparency. Nevertheless, one of its advantages is an extreme flexibility, therefore most of the basic components of Agile can still be effectively used by a single person. The key feature of the version of Agile adopted for the project was breaking down the project workload into clearly defined units of work, called iterations or sprints, and setting a milestone for each of them. Excel sheets were used for defining the set of tasks for each iteration. GitHub issue tracker was also used as a supporting productivity tool for this project. Code related tasks were transformed into issues within project GitHub repository. The issue tracker appeared to be a very efficient tool for keeping the focus.

This way of planning tasks was helpful in terms of the estimation of the sprint timescale (???).

There were several other techniques used for this project as a part of the adopted implementation of Agile methodology that are worth mentioning. First of all, it is TDD or test driven development. This means that unit tests were always written before the implementation and the next sprint has been started unless all test from the previous sprint passed. Always working on only one task from the sprint

In summary, in this section requirements from the analysis phase will be converted into the system design. First, several design cornerstones (website structure using Mind maps, wireframes,) were set in order to complete the initial prototype of the application. Other elements of the application design (use cases, database diagram) were updated in iterations inline with the Agile methodology.

4.1 Mind Map

After defining the set of use cases, I have created a block diagram of the future application. To accomplish this task, was used a program called MindJet enabling creation of a big scale mind maps. , project structure using mindset

The first step was made a block diagram of the project. I used a mind map as brainstorming tool taking the useless into consideration. It helped me to break my application down into separate webpages and blocks and to define the navigation between them.

4.2 Competitors

After all the important requirements and ideas were put on paper, it is worth taking a look at the potential competitors websites and application. This step is an important stage of an application prototyping process: it allows to learn from the best practices and possibly avoid potential errors.

Basically considered, of course, direct competitors. We studied not only the top social networking sites, but less popular. Delved into a huge reservoir of applications: all they had to learn to understand their behavior, how they communicate with the user, as they present the information.

We also analyzed the experience of indirect competitors. For example, for our section of music is iTunes and other specialized applications. When it comes to photos, then we photographed absolutely everything - in Instagram, Mobli etc. All this has given us an understanding of how to lead the audience in a variety of applications.

4.3 Wireframes

When speaking about prototyping, in the early stages the first choice of many designers is often a piece of paper and a pencil. Sketching has a number of advantages when compared to the use of the editors, such as Fireworks or Photoshop for prototyping. When using editors, it is easy to get distracted by brushing up unnecessary details too early. On the opposite side, sketches offer a lot of flexibility. It is easy to add notes, make small changes or replace an outdated sketch with a fresh one.

In case of this project, each of the sketches represented a separate view of the website. The scale of a view might differ. For example, some sketches represented a whole page (home page, dashboard page, etc.), others just outlined certain blocks of the website, such as header, footer, user profile container in more detail. I always added a lot of comments to explain the navigation and sometimes expected output. Sketches were one of the most powerful tools I used during the prototype process.

Scans of the drawings First website prototype in a photoshop/Fireworks Early prototype using html and css, using bootstrap should be quick and easy

4.4 Visual Design: Branding, icons, Font

Branding - number of names for the application Bootstrap was used as a framework on the front-end. Flat design is a big trend of the last years. It was decided to use the flat design in order to not distract user from the content. The design is a mixture of free templates and UI freebies created specifically for Bootstrap.

4.5 Use Cases

As mentioned above, use cases and the database diagram presented below, were prototyped in iterations. In this report will be presented a completed, merged version of the set of use cases and the database design. For use cases UML will be used to design in a clear and readable manner.

4.6 Database Design

what database was used Use this link to describe the ORM and its advantages: http://www.aosabook.org/en/sqlalchemy.html

4.6.1 Database Schema

Describe how the database was designed (what we need to capture and how I gradually added table by table). Start with user, as it is the cornerstone of the application (see forthergill) The database class diagram presented is a result of numerous iterations. Present also database before and after.

4.6.2 Database Class Diagram

Chapter 5

Implementation

From a development point of view the aim of this project was to create a working web application that also makes use of the best development practices, high quality architecture and is easy to maintain and extend in the future. The application is a relatively small-scale one, but it was developed with a future large-scale application in mind, that would allow many users to experience the application at once.

This chapter examines the implementation of the project.

5.1 Choice of Technologies

The application will be built using a set of front end and back end technologies.

5.1.1 Front End

The markup of the future application will be coded using HTML 5[1]. The application markup will be built using BEM front end development approach [2]. BEM (short for "Block Element Modifier") is a popular semantic model for markup and a way to organise sections of a website into purposeful blocks and to optimise CSS. The idea behind is to logically break the HTML down into *independent* blocks, which will allow arbitrary placement of the block anywhere on the page, including nesting the block inside another block. The approach can be very beneficial for large websites, allowing the code to be reused across pages or even projects. However, a small project like the SureThing can also benefit from BEM by making use of independent, context-free CSS that can be easily amended in the future[3].

CSS3 is used to define the visual presentation of the application. In general, CSS has certain limitations of its syntax capabilities. For example, it does not allow the use of variables, macros, mixins (reusable blocks of styles) functions and other features associated with object-oriented development, which inevitably leads to the creation of immensely repetitive stylesheets. In order to overcome those limitations, SASS preprocessor [4] will be used in this project. SASS (short for Syntactically Awesome Stylesheets) is a powerful language that extends CSS with a choice of useful functionality, all in CSS-compatible syntax. Use of SASS would allow to make CSS code more efficient and easily maintainable.

On the top of that, a popular CSS framework Bootstrap 3 [5] will be used. Bootstrap provides a number of ready solutions for designing the layout of the future application. Therefore, the overall architecture of the markup will be defined by identifying BEM blocks and elements. This would bring structure into the code across all front end technologies used in the development. BEM blocks and elements will be complemented with appropriate Bootstrap classes in order to speed up the development process and make the application fully responsive.

JavaScript, specifically JQuery library[6], will be used to add animations and improve overall user experience from using the application.

In order to handle time-consuming and repetitive tasks on the front end side, the task-based command-line tool Grunt will be used. This software comes with a variety of plugins serving different purposes. For this project will be used *grunt-sass* to compile SASS stylesheets into CSS complemented with *grunt-watch* to allow continuous development, *grunt-css* plugin to combine the all external CSS files into one and *grunt-uglify* plugin in order to reduce the size of JavaScript files and speed up loading of the web page in a browser.

In addition, RequireJS[7], a powerful asynchronous script loader will be used for effective management of JavaScript dependencies. It can load modules in asynchronous manner if desired and thus improve overall website performance.

5.1.2 Back End

For making reasonably accurate football results predictions the application requires latest footaball data. Live data would have to be frequently loaded into the system and processed in an appropriate way. Therefore, there would be a need for at least one separate module dealing with a third party football data API and containing business logic to manipulate the received data. The API wrapper is expected to be integrated into the web application, but separated from the presentation, it also has to be relatively easy to execute as a standalone module, encouraging a nicely decoupled design. Based on the above assumptions, Python was chosen as a primary back end language for this project being known as a language well suited to data manipulation.

The back end of the web application will be built using Python web framework Flask[8]. It is a lightweight framework (the official name is Python microframework) with a great choice of third-parties libraries (e.g. Flask-SQLAlchemy or Flask-Login) that can extend the functionality of the framework core in various ways. Flask application is minimalist to begin with, but it can grow with the project needs. For the purpose of this project this is an advantage compared to the frameworks like Django that have a big amount of functionality already built-in in the basic installation. In addition, availability of developer-friendly documentation and low learning curve makes Flask a short way to get a simple, Python-powered web site up and running. Threfore, Flask appears to be a great choice for a small project like SureThing.

SQLAlchemy was chosen as database solution for this project[9]. This is a powerful database framework that supports several databases back ends and offers the high-level Object Relational Mapper (for short, ORM). Using ORM provides a great level of abstraction when working with databases. For example, SQLAlchemy uses classes that map to each table in a database. This means that the records interaction can be kept the same regardless of the underlying database system. This offers a lot of

flexibility and, for example, allows to use different database systems for development and production environment. Furthermore, Flask-Migrate extension, based on a migration framework Alembic and written by a lead developer of SQLAlchemy, provides a powerful solution to handle database alterations and make database schema updates easily manageable[10].

(author?) [10]"dakdaslkd"

Cloud Deployment The latest trend in application hosting is to host in the cloud. This technology, which is formally known as Platform as a Service (PaaS), frees the application developer from the mundane tasks of installing and maintaining the hardware and software platforms on which the application runs. In the PaaS model, a service provider offers a fully managed platform in which applications can run. The application developer uses tools and libraries from the provider to integrate the application with the platform. The application is then uploaded to the servers maintained by the provider and usually is deployed within seconds. Most PaaS providers offer ways to dynamically scale the application by adding or removing servers as necessary to keep up with the number of requests received.

5.2 Application Architecture

own validation in forms own decorators fade out effect on alerts

5.2.1 Version Control

talk about labels annd how its cool that you can choose your own (edit a label, attach a screenshot) mentioned that you missed an option to set them as a list of priorities

5.2.2 Own Decorators

5.2.3 Custom Macros

Custom macros in JINJA templates

5.2.4 Third Party API

Many production Python web applications rely on external application programming interfaces (APIs). API can be also reffered to as "third party services" or "external platform".[11] For SureThing to work correctly, it requires constant access to live football data. After choosing an appropriate API, it has to be integrated into the application.

There is a variety of tools available for developers for accessing web APIs. Those three options were considered when choosing an appropriate tool:

- Helper library (such as Runscope or Apiary) Using a helper library has an overhead of learning how to use another piece of software.

- urllib2, standard Python module *urllib2* module offers very simple implementation and provides most of the required HTTP capabilities, but the API is thoroughly broken and features critical for performance are missing, for example connection re-using/pooling.
- urllib3
- requests, another Python library for handling HTTP requests. It offers a lot of control over the HTTP calls through the use of its powerful features.

After some experiments with other urllib2, urllib3 and requests, *requests* was chosen as the libarary for this project.

All interaction with the Football-API, including processing the received data, was separated out into a module *football_api_wrapperpy* or just "wrapper" for short. This module contains only one class, FootballAPIWrapper. Fields of the class accomodate the key elements of the interaction with the API that will be re-used in different methods of the wrapper, for example base url, path to the data directory.

```
def __init__(self):

    self._premier_league_id = '1204'
    self._base_url = 'http://football-api.com/api/?Action='
    self._basedir = os.path.dirname(__file__)
    self._datadir = os.path.abspath( os.path.join(self._basedir, '..', 'data') )
    self._proxy_on = False
```

Figure 5.1: Football API wrapper, fields

To get responce from Football-API takes about 11s, therefore it is needed to move the API calls into a task queue so they do not block the HTTP request-response cycle for the rest of the web application.

5.3 Features Implementation

In this section will be described the technical details of the project implementation. Each susbsection is bound to the high level feature of the application, as indroduced in the chapter "Requirements Analysis" 3.4.2.

5.3.1 Authentication

The application requires authentication functionality.

5.3.2 Matches Overview

5.3.3 Prediction

describe the way prediction settings apply, three levels of prediction settings how the overall prediction is calculated, weighted prediction

5.3.4 Upcoming Match View

Implementation of this view was one the most complex development tasks of the whole project.

5.3.4.1 User journey

The user can navigate to this view either from the matches overview on the main page or from the dashboard (if the match has already been saved to the dashboard).

If the user is coming to the upcoming match preview from the *main page*, it will display the match header (containing general information about the teams, last played game, match kick-off time and date, etc.), a list of prediction modules and a *prediction value* for each module calculated based on the relevant statistics for each of the teams. This information should be sufficient for the user to decide, whether it is worth saving the match to the dashboard for a latter revision. An unauthorised user would be able to see the same overview, but the "Save" button will be disabled.

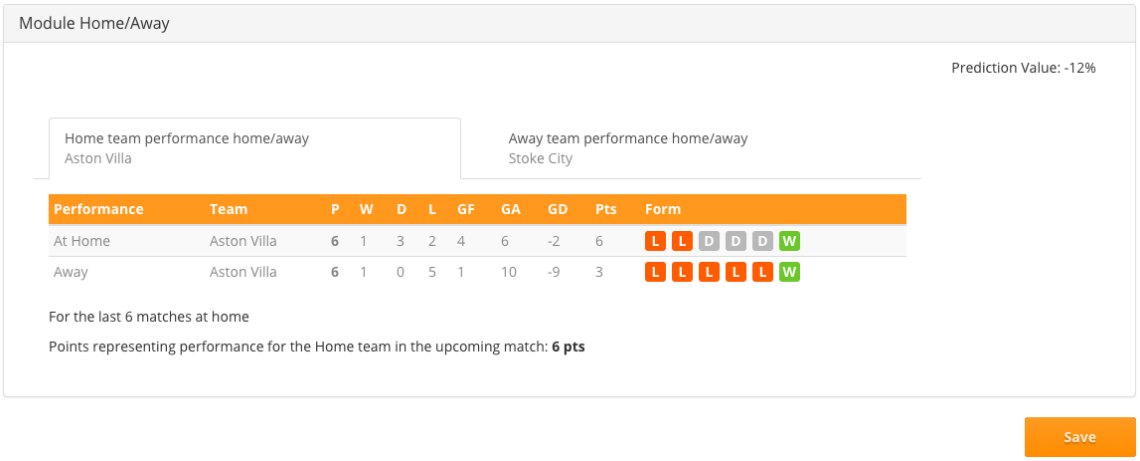

Figure 5.2: An example of a prediction module in the Match Preview, user navigated from the main page.

If the user comes to the upcoming match preview from the *dashboard*, they will be able to see more information related to the actual result prediction and betting. First of all, in each prediction module they will see an input fields for setting match specific prediction weights. Secondly, they will see a user hunch module. Finally, at the very bottom of the overview they will see calculated prediction result and two buttons - one to save the match specific settings and another to commit the bet.

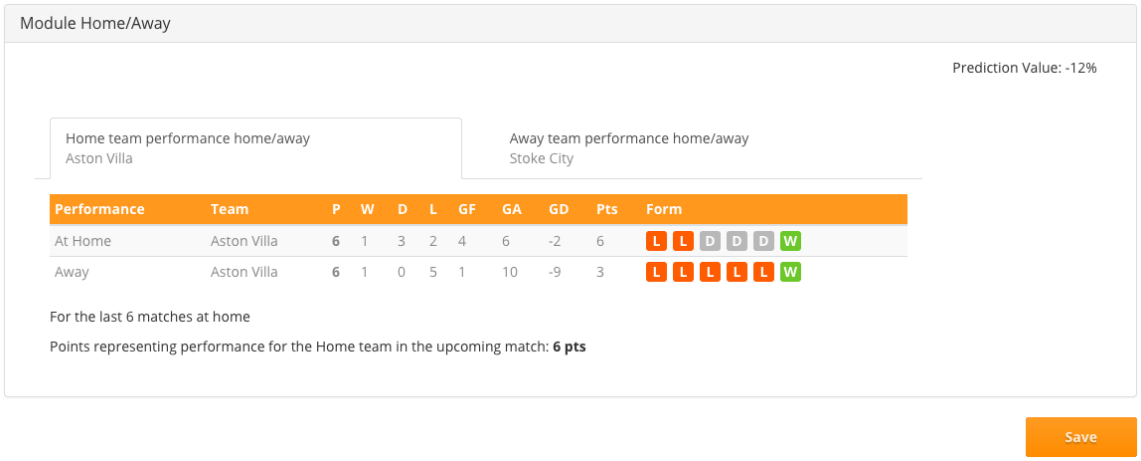

Figure 5.3: An example of a prediction module in the Match Preview, user navigated from the dashboard.

	Prediction values	User weights	User weights incl. user hunch	Prediction excluding user hunch module	Prediction including user hunch module
Module league position	78%	20%	20%	15.6%	15.6%
Module form	44%	30%	20%	13.2%	8.8%
Module home/away	77%	70%	10%	53.9%	7.7%
Module user hunch	-100%		50%	x	-50%
Match result prediction				82.7%	-17.9%

Figure 5.4: Table, user hunch influences prediction result

	Prediction values	User weights	User weights incl. user hunch	Prediction excluding user hunch module	Prediction including user hunch module
Module league position	78%	20%	20%	15.6%	15.6%
Module form	44%	30%	20%	13.2%	8.8%
Module home/away	77%	70%	10%	53.9%	7.7%
Module user hunch	-100%		50%	x	-50%
Match result prediction				82.7%	-17.9%

Figure 5.5: Table, user hunch influences prediction result

most Explain how was implemented user hunch: combination of Flask Ajax and Sockets IO!!!

5.3.4.2 Possible Enchancement

Although, one of the key features of the application is to try not to overhelm user with statistics, as opposed to many football stats websites, this view would need a little bit more additional information to complete the big picture.

5.4 Application Performance

sockets, threads multithreading how I fixed performance on Match.update_all_matches

5.5 Factory Pattern

Application Factories The factory pattern is the first pattern to be implemented and used in any of my Flask applications. There is a small amount of documentation regarding application factories already. While the documentation is limited in scope, I believe it is there to encourage the usage of this pattern. That being said, there is not an established convention for implementing a factory method. Chances are your app will have its own unique requirements and thus your factory method should be tailored accordingly. Regardless of your implementation the factory method is, in my opinion, indispensable as it gives your more control over the creation of your application in different contexts such as in your production environment or while running tests.

Within the Overholt source code you will find three different factory methods. There is one factory for each application and an additional factory which is shared by the individual application factories. The shared factory instantiates the application and configures the application with options that are shared between apps. The individual app factories further configure the application with options that are more specific to thir use. For example, the api application factory registers a custom JSONEncoder class and custom error handlers that render JSON responses. Whereas the frontend application factory initializes an assets pipeline and custom error handlers for HTTP responses.

5.6 Blueprints

Blueprints are crucial to my Flask applications as they allow me to group related endpoints together. I honestly couldn't live without Blueprints. The Flask documentation provides the best overview of what Blueprints are and why they are useful. There isn't much else I can describe about Blueprints themselves that Armin hasn't already. In the context of the Overholt source code, each application package contains various modules containing Blueprint instances. The API application contains three Blueprints located at overholt.api.products, overholt.api.stores and overholt.api.users. The frontend application contains but one Blueprint located at overholt.frontend.dashboard. All Blueprint modules are located in the same package as the application which allows me to use a simple method of registering them on their respective application. Within the shared application factory you should notice the register_blueprints helper method. This method simply scans all the modules in the application package for Blueprint instances and registers them on the app instance.

5.7 Configuration

Configuration is always important for an application, especially for sensitive details such as API keys and passwords. I always provide a default configuration file that is checked into the project repository so that a developer can get up and running as quick as possible. This file contains default values that are specific to the virtual machine settings specified in the Vagrantfile. This default file is used to configure any apps created by the shared application factory. Additionally, the factory method attempts to override any default settings from a settings.cfg file located in the application's instance folder. Head over here for more information regarding Flask's instance folders. This additional file can be created by any developer working on the project to tweak any settings to be more specific to their

local development environment. When it comes time to deploy the application to a development or production server the settings.cfg file will be created by the deployment tool, such as Chef or Fabric.

5.8 Database Migrations

In addition to using SQLAlchemy I always use Alembic. Alembic is a nice database migration tool made specifically for SQLAlchemy by Mike Bayer, the author of SQLAlchemy. What's nice about Alembic is that it includes a feature to autogenerate database versions from the model metadata. If you examine the alembic.env module you should notice the application specific imports. Further down is where the application's database URI and model metadata is handed off to Alembic. I've written previously about using Alembic with Flask and I would suggest reading that article for a little more detail.

5.9 RequireJS

5.9.1 Deploying the Application

Cloud Deployment is the most recent trend in application hosting. The formal name of this technologu is Platform as a Service (PaaS). In the PaaS model, a service provider offers a fully managed platform in which applications can run.

The application will be hosted on "The latest trend in application hosting is to host in the cloud. This technology, which is formally known as Platform as a Service (PaaS), frees the application developer from the mundane tasks of installing and maintaining the hardware and software platforms on which the application runs. In the PaaS model, a service provider offers a fully managed platform in which applications can run. The application developer uses tools and libraries from the provider to integrate the application with the platform. The application is then uploaded to the servers maintained by the provider and usually is deployed within seconds. Most PaaS providers offer ways to dynamically scale the application by adding or removing servers as necessary to keep up with the number of requests received. Cloud deployments offer great flexibility and are relatively simple to set up, but of course all that goodness comes at a price" Flask applications

5.10 Conclusions

The main conclusions for this chapter.

Chapter 6

Evaluation & Testing

This chapter evaluates the overall project and provides results of tests carried out.

6.1 Conclusions

The main conclusions for this chapter.

Chapter 7

Conclusion

This chapter summarises the main outcomes and conclusions resulting from this body of work.

7.1 Assessment of Success

There are meny aspects of this project that should be inspected in order to decide if the project was successful or not.

Using Agile development approach allowed me to concentrate on the result without wasting my time creating final diagrams and overviews before starting with the implementation. It helped me to cut down the preparation phase to the bare essentials, reduced project overhead and made the whole development process more efficient.

Overall, I feel the project has been very successful with a well-designed web application as a result.

7.2 Improvements and Future Work

Further development that could be carried out in the future. In the future I expect to put more leagues, so any league is supported More modules (Injuries and suspensions, manager) remove a module from the view more detiled information in the played match, a view containing feedback

Chapter 8

Using LaTeX

There are several reasons why one should prefer LaTeX to a WYSIWYG word processor like Microsoft Word: *portability*, *lightness*, *security* are just a few of them (not to mention that LaTeX is free). There is still a further reason that should definitely convince you to abandon MS Word for the development of a dissertation: you will never be able to produce professionally typeset and well-structured documents using most standard WYSIWYG tools.

LaTeX is a free typesetting system that allows you to focus on content without bothering about the layout: the software takes care of the actual typesetting, structuring and page formatting, producing documents of astonishing elegance.

8.1 Structure of this Template

The file thesis.tex in the root of the directory (ThesisTemplate) is the main file of this template. This is the file that must be compiled to create the document. The thesis.tex document contains a lot of configuration settings. The only elements that require editing are details such as the title of the report, authors name and so forth. The only further addition to the file is to use the *include* statement to include additional chapters into the report. One may also comment out the *include* statements using the percentage sign (%) to develop the report on a chapter by chapter basis. The BibTeX database thesis.bib is also included within the root. All the actual content of the report is divided up into directories each with a .tex file containing the chapter content.

8.2 Using Figures

One can insert graphic elements using LaTeX in a number of ways. Vector based imagery such as diagrams saved to the pdf format may use the *includegraphics* command with the optional *viewport* attribute to specify a precise area of the graphic to be included. Figures should also include a Caption and a Label for referencing.

When inserting a Figure (Figure 8.1) one uses the *begin*{*figure*} and *end*{*figure*} commands. The image presented is a vector graphic in the form of a pdf file. When working with such files it is usually necessary to include the optional *viewport* attribute to designate the specific area in which to focus.

The first pair of coordinates (x & y) designate the pixel location of the lower left corner. The second pair identify the upper right hand corner. Modification of these coordinates allows one to focus in upon a particular area of interest within the vector image. The optional attribute [H] when beginning a Figure inserts the graphic element at the specified location. Other options such as [htb] (here, top, bottom) will place the graphic in the most suitable place that LaTeX can find. This however can have a negative impact on memory allocation if a large number of images are to be found within the document.

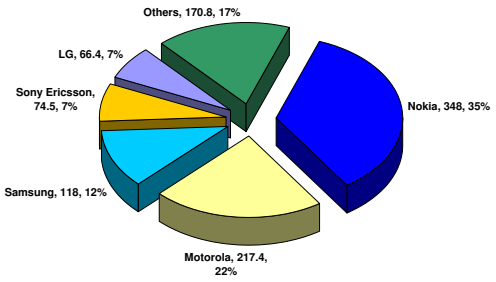

Figure 8.1: Mobile Phone Suppliers, Market Share 2006 (Millions of Units Shipped)

One may use the *minipage* command when inserting two figures to span across the page. This allows for the subdividing of the page into a number of columns of specified width. Note the pie-charts presented here (Figure 8.2 & 8.3) are a bit small for viewing as printed matter. Zoom in on them using a pdf reader to see the advantage of printing diagrams (vector graphics) in pdf format.

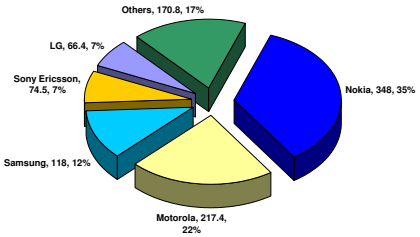

Figure 8.2: Caption for Figure

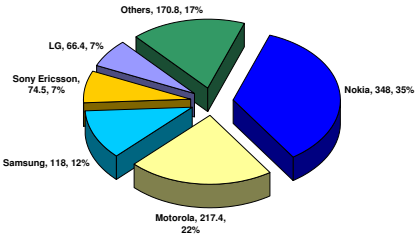

Figure 8.3: Caption for Figure

The example below (Figure 8.4) demonstrates the insertion of a bitmap image. One can see that the extension for the image file isn't specified, as this template is setup to automatically search for .jpg, .png, .gif and .pdf images. The size of the displayed image within the document can be varied by adjusting the height and width attributes. To rotate an image 90 degrees an optional attribute can be added, for example [width=.4\linewidth, angle=90].

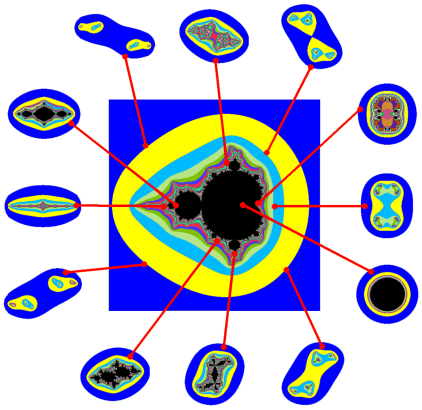

Figure 8.4: Caption for Bitmap Image Example

8.3 Referencing

To refer to another part of the document one must use a combination of the *ref* and *label* commands. The label is a unique identifier, therefore when working with large documents it helps to give references meaningful names. Examples of this includes prefixing Table references with *tab:*, figures with *fig:*, chapters with *ch:*. In very large documents in may also be useful to add an additional level of prefixing to represent the chapter the label is in. In this example chapter the tables and figures have the additional prefix of *using* to represent the *usingLatex* chapter. The tilde (~) is used to ensure that a reference remains as a single object. All instances of *ref* should be preceded with the tilde.

8.4 Citing Bibliographic References

Bibliographic references are stored in a database (.bib file extension), this contains a list of articles, proceedings, books, thesis and so forth. Each type of publication has a number of required fields such as a unique identifier, author and title. To cite a references within the main body text one must use the *cite* command as in the following examples. Albert Einstein [**?**] for example is well known for his work on Brownian Movement. The SETI@Home project [12] is an example of a webpage citation. One can also work with articles [13], MSc Thesis [14], PhD Thesis [15] or articles within conference proceedings [16]. Several other types of article exist, but they are used to a lesser degree.

8.4.1 Compiling a BibTeX Database

Having initially compiled the document using pdfLatex a number of helper files are created that aid in referencing and citations. One then must compile the bibtex database, followed by an additional two compiles using pdfLatex. Citing additional bibliographic references within the body of the document being produced will require the recompile of the bibtex database. In the case that the bibtex reference of a cited article cannot be found one will see a question mark [?] instead of the proper citation.

8.4.2 The Vancouver Bibliography Style

The bibliographic references are laid out using the Vancouver style. Further information about this style may be found at the following link (http://gentoo.chem.wisc.edu/tex-archive/biblio/bibtex/contrib/vancouver/).

8.5 Working with Tables

The data in a table (Table 8.1) displays four columns of left-aligned data. The cell contents can be aligned to the left (l), right (r), or center (c). Vertical bars may sometimes be seen in tables but these generally look unprofessional. The Booktabs package [17] allows for the creation of professional looking tables as shown in the example. The use of *\toprule*, *\midrule* and *\bottomrule* commands provided by the package allow for rules of varying thickness and spacing. Data elements (cells / columns) within a table are divided up using the ampersand (&). To complete a row one must end with a double backslash (\\). Tables as with figures need a caption and a label.

Heading 1	**Heading 2**	**Heading 3**	**Heading 4**
Cell A1	Cell B2	Cell C3	Cell D4
Cell E1	Cell F2	Cell G3	Cell H4

Table 8.1: Table Caption

One may find the following spreadsheet tool (http://cobweb.ecn.purdue.edu/~zhang97/xls2latex/) to be particularly useful for quickly converting tabular data in a spreadsheet to LaTeX form. A complete list of instructions on how to use the tool are also present. The WinShell editor has an in-built GUI based utility to aid in the creation of the tabular data.

8.6 Inserting an Algorithm

The *algorithm2e* environment may be used to generate algorithms (Algorithm 1). The following document (http://www.tex.ac.uk/ctan/macros/latex/contrib/algorithm2e/algorithm2e.pdf) provides detail of all the commands available within the package. If no algorithms are used within the document then comment out line 70 of the file `thesis.tex` to remove the list of algorithms from the contents area.

Algorithm 1: A Sample Algorithm

```
while (RANK < COMPSIZE) do
    if (RANK == MASTER) then
        generate random value ;
        for (each item K) do
            get result ;
        end
    end
end
```

8.7 Inserting Program Code Samples

To insert small segments of program code (Listing 8.2) that detail how interesting algorithms and so forth are implemented use the *lstlisting* command. Inclusion of program code again requires a Caption and Label. Sample code from external files may also be included (Listing 8.1), again one must supply a Caption and a Label as well as the relative path to the source file. Note a paragraph of text consisting of just a few lines is not a paragraph.

```
1 public void someInterestingMethod( int x ) {
2   try {
3     DataInputStream in = new DataInputStream(req.getInputStream());
4     HttpSession session = req.getSession(true);
5     //Read Input Data
6     in.close();
7   } catch ( Exception err ) { }
8   outputStream.close();
9 }
```

Listing 8.1: The Caption for the Code Listing

```
1 if(rndVal==0){
2     if(opType > 2){
3        //do something
4     }
5 }
```

Listing 8.2: Sample Program Code Listing

8.8 Working with Maths

One can insert mathematical formula directly into a paragraph of text. The mathematical definition of the "Cantor set" is a good example of this in action $\sum_{n=0}^{\infty} \frac{2^n}{3^{n+1}} = \frac{1}{3}+\frac{2}{9}+\frac{4}{27}+\frac{8}{81}+\cdots = \frac{1}{3}\left(\frac{1}{1-\frac{2}{3}}\right) = 1.$ The previous equation demonstrates the use of sigma, fractions, large brackets, power, and dots. The function that defines the MSet is a simpler example of math in use $Z_{n+1} = Z_n^2 + C$. Matrix Multiplication is typically regarded as an $O(n^3)$ operation. One may use the *equation* environment for more complex mathematical formula that should standout. One may take for example the product C of two matrices $A \in M_{n,m}(R)$ and $B \in M_{m,p}(R)$ to be defined as

$$(A \times B)_{ij} = \sum_{k=0}^{m-1} a_{ik} b_{kj}, \ i = 0, ..., n-1, \ j = 0, ..., p-1. \tag{8.1}$$

The sizes of the matrices must satisfy $(n \times m)(m \times p) = (n \times p)$. Matrix multiplication is an associative process thereby $a \cdot (b \cdot c) = (a \cdot b) \cdot c$. Essentially to find the value of a particular cell $C_{i,j}$ it is necessary to multiply row i of the matrix A with column j of matrix B summing all the multiplications.

8.9 Required Software

The implementation of LaTeX typically used is MikTeX (http://miktex.org/). It is best to install the complete MiKTeX system. Initially a small installer application must be downloaded and executed. This in-turn downloads the most recent implementation of the MikTeX system. The complete system is circa 500MB in size. Run the installer again and select the directory of the downloaded package.

To allow one to work with postscript documents a system called Ghostscript in required (http://pages.cs.wisc.edu/~ghost/). To be able to view these documents download and install GhostView (http://pages.cs.wisc.edu/~ghost/gsview/index.htm). The last element that is necessary is an editor. TeXnicCenter is a free download available at (http://sourceforge.net/projects/texniccenter/). An alternative is WinEdt a shareware ASCII editor (http://www.winedt.com/). WinEdt can be freely used for a 30 day period, after which one will periodically receive reminders to register the product. To demonstrate the use of an enumerated list, the software should be installed in the order detailed below. One final option for Windows users is WinShell (http://www.winshell.de/). An advantage of WinShell is its in-built BibTeX GUI editor. It also features a useful Table Wizard.

1. MikTeX
2. Ghostscript
3. Ghostview
4. Editor

8.10 Working with Quotes

To surround a piece of text with double quotes one must place two single quotes on either side of the text. The double quote on the left is created using two left quotes (‘) this is located just above the *tab* key on the keyboard. The right hand double quote is created using two right hand quotes via (’) located just above and to the left of the right shift key. A properly formatted quotation should look like “This is a quotation”. Notice how the direction of the quotes are opposite to one another.

8.11 Further Information

The Not so Short Introduction to LaTeX (http://tobi.oetiker.ch/lshort/lshort.pdf) is one useful source of further information on how to work with this system.

8.12 Conclusion

A short conclusion providing a summary of the chapter.

Appendix A

Wireframes

When the app is launched it silently registers with the server allowing the user to use the app immediately. The user is then shown the middle-top screen.

1. From the middle top screen, the user can follow arrow 1 by clicking on the middle button "Pick Mission" to pick a Mission (Run around Arran or Egg for example) and then pick a start and end location. After confirming these choices, the user is taken back to the middle top screen, or at any time can click the "Home" button to return.

2. The user can also view their current acheivements by clicking the "Achievements" button on the middle top screen, following arrow 2. These achievements will be grouped by tabs by category - Distance, Time, Stage and Mission based achievements.

3. The user can follow arrow 3 from the middle top screen to notify the app that they are starting an exercise period, telling the app to track their distance. If a Mission and start and end location are not picked (as in point 1) then they will instead be redirected to this screen and are unable to start exercising until this choice has been made. Once they have successfully advanced to this screen, it will display their current progress as they move showing the user how close to completion of their current stage and overall route they are.

4. When the user has finished exercising, they will click the "End Session" button and be taken to the first summary screen - following arrow 4. Here statistics from their exercise will be shown and the option to share this on several social media outlets.

5. The user can then move to the second and final summary screen, following arrow 5, where they will be shown any achievements they were awarded during that session. The user will also have the option to share these on social media outlets. From here, the user can click the "Home" button and be taken back to the middle top screen.

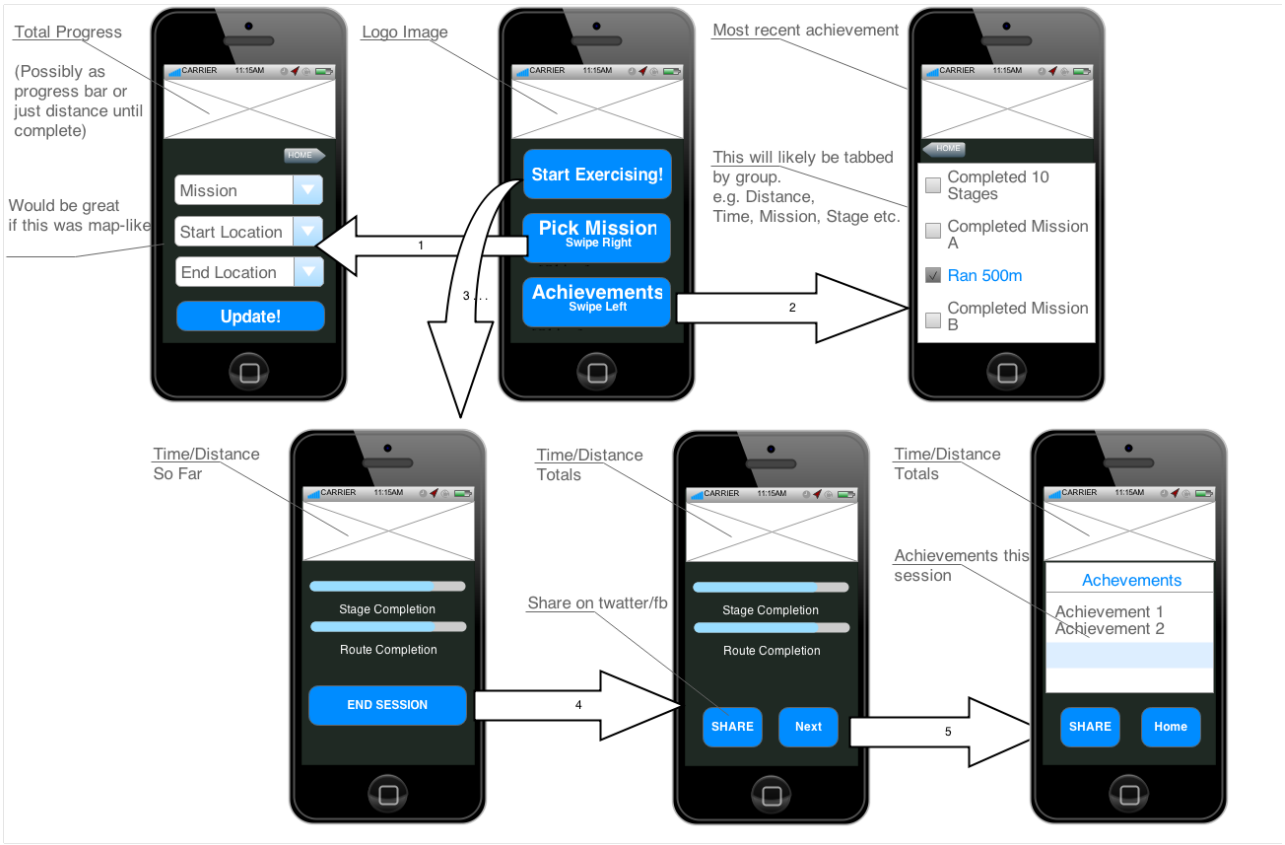

Figure A.1: Wireframes, initial design

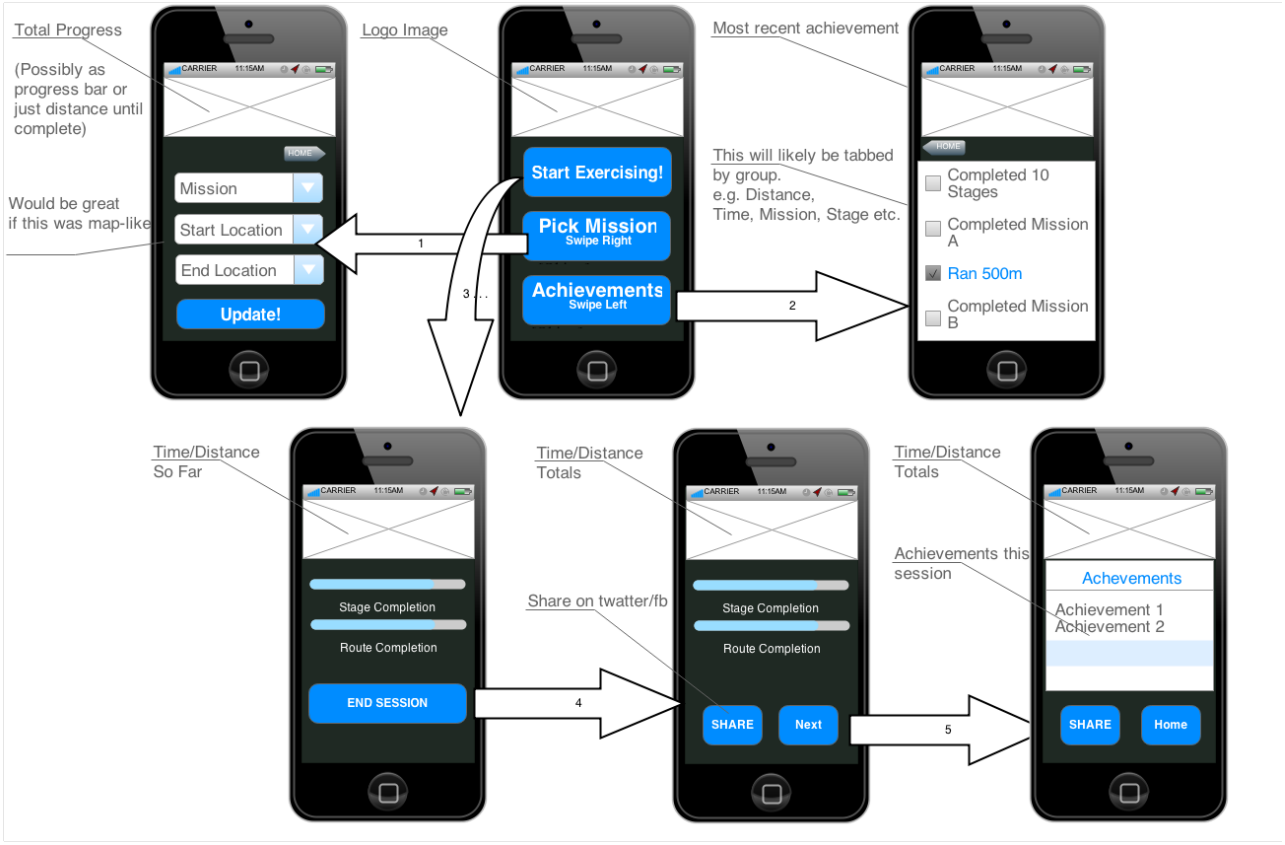

Figure A.2: Inital sketch of wireframe ideas

Appendix B

Installation Instructions

The code can be checked out using git by executing the following command in the terminal:

See the following command :

```
1  $ git clone git@github.com:marinamarina/sure-thing.git
```

Installation instructions are found at the following url:

https://www.github.com:marinamarina/sure-thing/blob/master/README.md.

If any issues arise regarding installation of any part of the system, do not hesitate to contact me at

1014481@rgu.ac.uk

Appendix C

Project Specification

Summary of the project outline.

C.1 Functional Requirements

some text here

C.2 Non-Functional Requirements

some text here

Appendix D

Project Management

Discussion on how the project was managed. What things impacted the success of the project. How does the continually revised versions of the project plan compare to the initial draft developed at the start of the project. Did everything run according the schedule. Did elements such as exams & coursework have any impact.

Appendix E

Another Appendix

This appendix makes use of the *rotating* package to rotate both figures and tables ninety degrees allowing for large datasets and illustrations to be represented.

Heading 1	**Heading 2**	**Heading 3**	**Heading 4**	**Heading 5**	**Heading 6**	**Heading 7**	**Heading 8**	**Heading 9**	**Heading 10**
aaa	bbbb	cccc	dddd	eeee	ffff	gggg	hhhh	iiii	jjjj
aaa	bbbb	cccc	dddd	eeee	ffff	gggg	hhhh	iiii	jjjj
aaa	bbbb	cccc	dddd	eeee	ffff	gggg	hhhh	iiii	jjjj
aaa	bbbb	cccc	dddd	eeee	ffff	gggg	hhhh	iiii	jjjj
aaa	bbbb	cccc	dddd	eeee	ffff	gggg	hhhh	iiii	jjjj
aaa	bbbb	cccc	dddd	eeee	ffff	gggg	hhhh	iiii	jjjj

Table E.1: A much longer caption that will not be listed in the list of tables page.

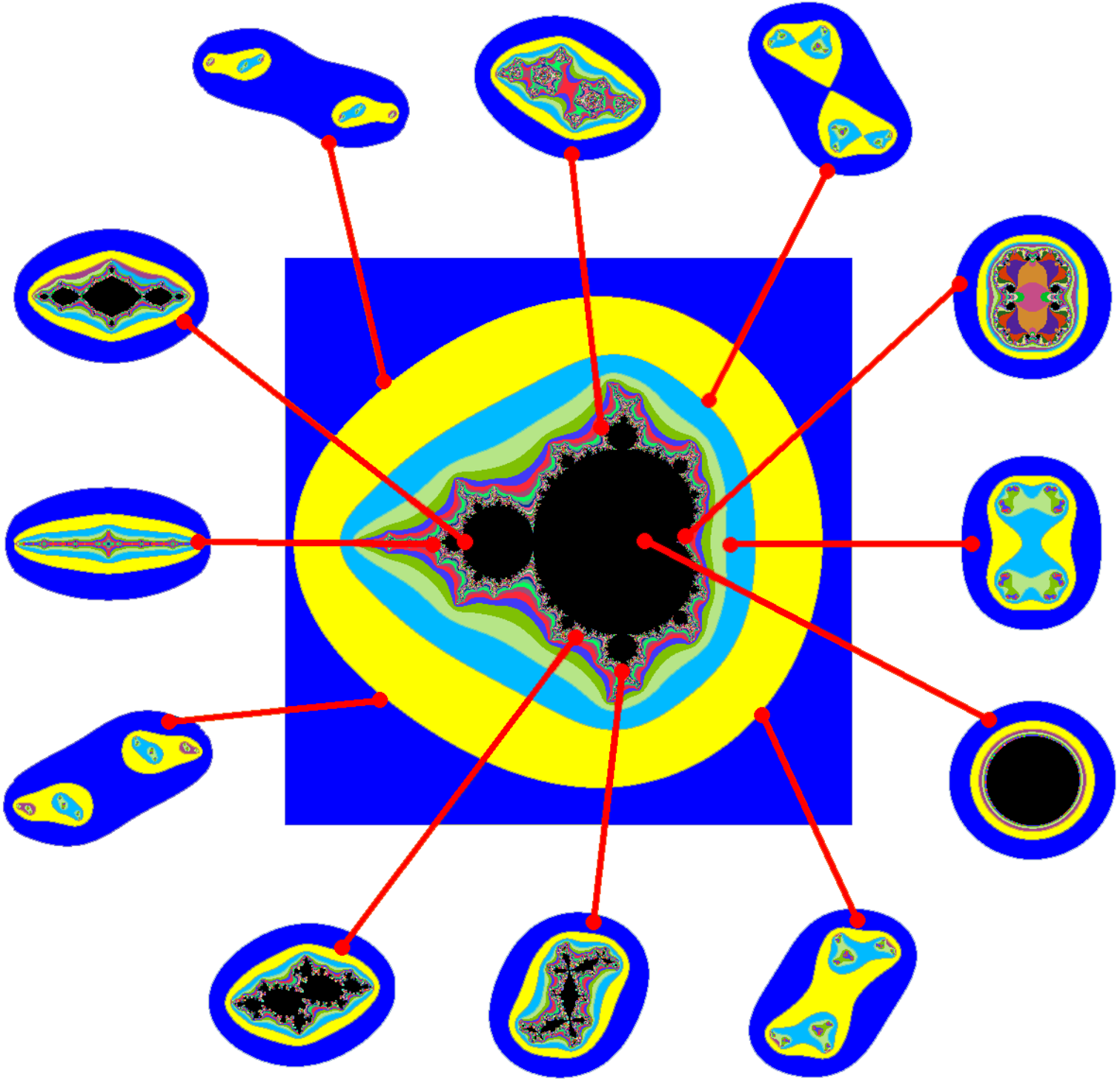

Figure F.1: A much longer caption that will not be listed in the list of figures page

Appendix F

Implementation

Third-Party Code and Libraries

The slides from the formal presentation should be provided here in not more than two pages.

Appendix G

Project Log

The following is a weekly summary of the work carried during the development of this body of work. It covers tasks that were completed, tutorials that were worked through, articles that were read and reviews of discussions / meetings held with the project supervisor and other third parties.

Week Beginning: Monday 27/09/2010

First week working on the project. Had a meeting with supervisor and discussed some of the issues related to the project. The first deliverable is due for the end of next week (project outline & ethics form).

- Downloaded and Installed LaTeX (MikTeX full install), Ghostscript, Ghostview & Winshell.
- Started to get to grips with the LaTeX system by making simple modifications to the template and editing the project log.
- Developed a Mind Map to clarify understanding of project elements.
- Prepared an initial draft of project plan in the form of a Gantt chart.
- Prepared and revised 1 page draft of project summary & filled in ethics form.

Bibliography

[1] WHATWG W. HTML (HyperText Markup Language); 2014. http://www.w3.org/html/.

[2] Yandex. BEM Methodology; 2005. http://https://bem.info/.

[3] Shirshin M. Scaling Down The BEM Methodology For Small Projects; 2014. http://www.smashingmagazine.com/2014/07/17/bem-methodology-for-small-projects/.

[4] Hampton Catlin CE Natalie Weizenbaum, various contributors. SASS Preprocessor; 2015. http://sass-lang.com/.

[5] Twitter. Twitter Bootstrap CSS3 Framework;. http://getbootstrap.com.

[6] Resig J. jQuery;. http://jquery.com/.

[7] tobeadded. RequireJS, JavaScript file and module loader;. http://requirejs.org/.

[8] Ronacher A, various contributors. Flask Micro Framework; 2014. http://flask.pocoo.org/.

[9] Bayer M, various contributors. SQLAlchemy ORM; 2014. http://docs.sqlalchemy.org/en/rel_0_9/orm/.

[10] Grindberg M. Flask Web Development. Developing Web Applications with Python. OReilly Media; 2014.

[11] Makai M. API Integration; 2015. http://www.fullstackpython.com/api-integration.html.

[12] Berkeley University. Berkeley Open Infractructure for Network Computing; 2010. Available from: http://boinc.berkeley.edu/ [cited 20 Sept 2010].

[13] Akenine-Möller T, Ström J. Graphics for the Masses: A Hardware Rasterization Architecture for Mobile Phones. ACM Trans Graph. 2003;22(3):801–808. Available from: http://www.cs.lth.se/home/Tomas_Akenine_Moller/pubs/masses.pdf.

[14] Edberg T. Evaluation and Optimisation of the SMILE Fluxgate Magnetometer [MSc Thesis]. Royal Institute of Technology, Stockholm, Sweden; 2007. Available from: http://www.ee.kth.se/php/modules/publications/reports/2007/XR-EE-SPP_2007_001.pdf.

[15] Grace P. Overcoming Middleware Heterogeneity in Mobile Computing Applications [PhD Thesis]. Computing Department Lancaster University; 2004. Available from: http://www.lancs.ac.uk/postgrad/gracep/thesisfinal.pdf.

[16] Park JH, Kim KO, Jeon JW. PDA based User Interface Management System for Remote Control

Robots. In: SICE-ICASE, 2006. International Joint Conference; 2006. p. 417–420. Available from: 10.1109/SICE.2006.315333.

[17] Fear S. Publication Quality Tables in LaTeX; 2005. Available from: http://www.ctan.org/pub/tex-archive/macros/latex/contrib/booktabs/booktabs.pdf [cited 20 Sept 2010].

[18] International E. ECMAScript Language Specification. 5th ed. ECMA International; 2011.

[19] Crockford D. JSON (JavaScript Object Notation);. http://martinfowler.com/articles/continuousIntegration.html#PracticesOfContinuousIntegration.

[20] Bocoup OWCD. The JavaScript Task Runner;. http://gruntjs.com/.

[21] Opta Sports; 2015. http://www.optasports.com.

[22] Zhou H, Mao K. Adaptive Successive Erosion-based Cell Image Segmentation for p53 Immunohistochemistry in Bladder Inverted Papilloma. Conf Proc IEEE Eng Med Biol Soc. 2005;6.