Football Betting Application

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 project aims to build a web application simulating football betting experience, addressing two main issues. Firstly, filling an existing void of an application that makes football match prediction customisable and transparent to the user. Users of the web application would be able to create their own betting system by adjusting the prediction percentage of several factors that may influence the output of the match. Secondly, allowing the users to analyse their past performance and compare their results and prediction settings with the settings of other users of the application.

The stated above would be achieved by taking several steps. On completion of a background research, current football prediction web applications will be researched and a set of requirements will be produced 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, a working web application will be produced and 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 daughter Scarlett for their support and infinite 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

When 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 engaged in football betting on a regular basis. 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 F.

3.3 Researching the Potential competitors

Before gathering the project requirements, it is a good practice to conduct research on what current systems with similar features are already available to users, looking at pros and cons of each of them. In case of a web application it means researching and analysing websites and mobile application. The research can be a source of inspiration and would also help to avoid possible design mistakes. It is important to analyse a large variety of applications in order to understand their main purpose, the way the applications present information and communicate with its users.

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 additional set of requirements. Therefore, the Requirements analysis is a crucial part of the project and it is the cornerstone of the software development life cycle. This phase is especially important when developing an application for a client, as gathering clear and specific requirements would help to ensure that both sides understand and agree on the feature set of the future application have.

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

3.4.1 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.

3.4.1.1 General Web Application Requirements

These are the requirements related to the basic functionality of the web application.

- 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
- the application will provide an option to change user password for security reasons

3.4.1.2 Matches Overview

- on the main page user will be able to see an overview of all matches for the current season

- using simple navigation tabs user should be able to switch between upcoming and played matches
- for each of the unplayed matches user can see more details about the match including the output of each of the prediction modules
- user is able to save an unplayed match to the dashboard
-

3.4.1.3 Match View

- hello
- from the prediction module user can see stats of the website population

3.4.1.4 Dashboard

- 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.

3.4.1.5 Prediction'

These are the requirements related to the basic functionality of the web application.

- the application should allow users to register and create a new account with the application

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

3.4.2 Non-functional Requirements

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

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 the project workload down into clearly defined blocks of work, called iterations or sprints, and setting a milestone for each of them. Excel sheets were used for defining the set of prioritised tasks for each iteration. Code related tasks were also added to the GitHub repository of the project as a set of issues attached to the current milestone. Planning tasks in this way 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 Task list at a time also appeared very efficient for keeping the focus.

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

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

This chapter examines the implementation of the project.

own validation in forms own decorators fade out effect on alerts

”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

5.1 Semantic HTML

While semantics is considered the foundation of web development. However, the HTML of a modern app is mostly a div soup. CSS by itself does not offer any structured model at all.

5.2 Why Framework?

just because

5.3 Flask

It has a rich ecosystem of plugins and snippets and is of course open source! Flask is ideal for building lightweight applications or APIs. It is also very easy to learn.

5.4 Macros

5.5 Performance

multithreading how I fixed performance on Match.update_all_matches

5.6 Git

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

5.7 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.

5.8 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.

5.9 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.

5.10 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.

5.11 RequireJS

5.12 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 Conclusions

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.

7.2 Future Work

Further development that could be carried out in the future.

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.

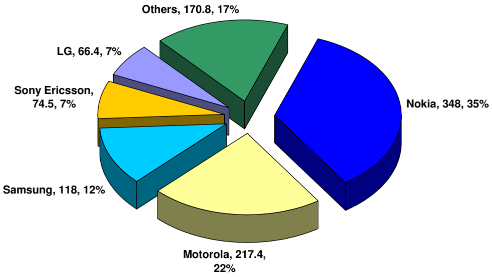

Figure 8.1: Mobile Phone Suppliers, Market Share 2006 (Millions of Units Shipped)

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.

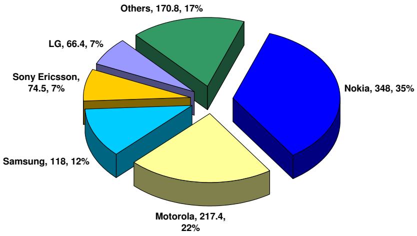

Figure 8.2: Caption for Figure

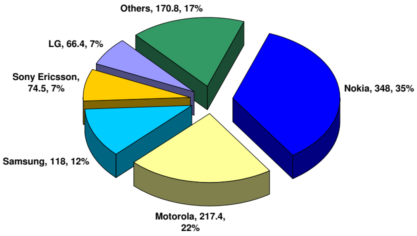

Figure 8.3: Caption for Figure

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].

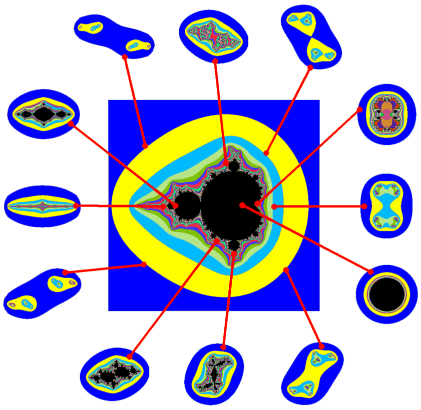

Figure 8.4: Caption for Bitmap Image Example

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 [1] for example is well known for his work on Brownian Movement. The SETI@Home project [2] is an example of a webpage citation. One can also work with articles [3], MSc Thesis [4], PhD Thesis [5] or articles within conference proceedings [6]. 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 [7] 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.

```
public void someInterestingMethod( int x ) {
  try {
    DataInputStream in = new DataInputStream(req.getInputStream());
    HttpSession session = req.getSession(true);
    //Read Input Data
    in.close();
  } catch ( Exception err ) { }
  outputStream.close();
}
```

Listing 8.1: The Caption for the Code Listing

```
if(rndVal==0){
    if(opType > 2){
       //do something
    }
}
```

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

Project Specification

Summary of the project outline.

A.1 Functional Requirements

some text here

A.2 Non-Functional Requirements

some text here

Appendix B

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 C

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

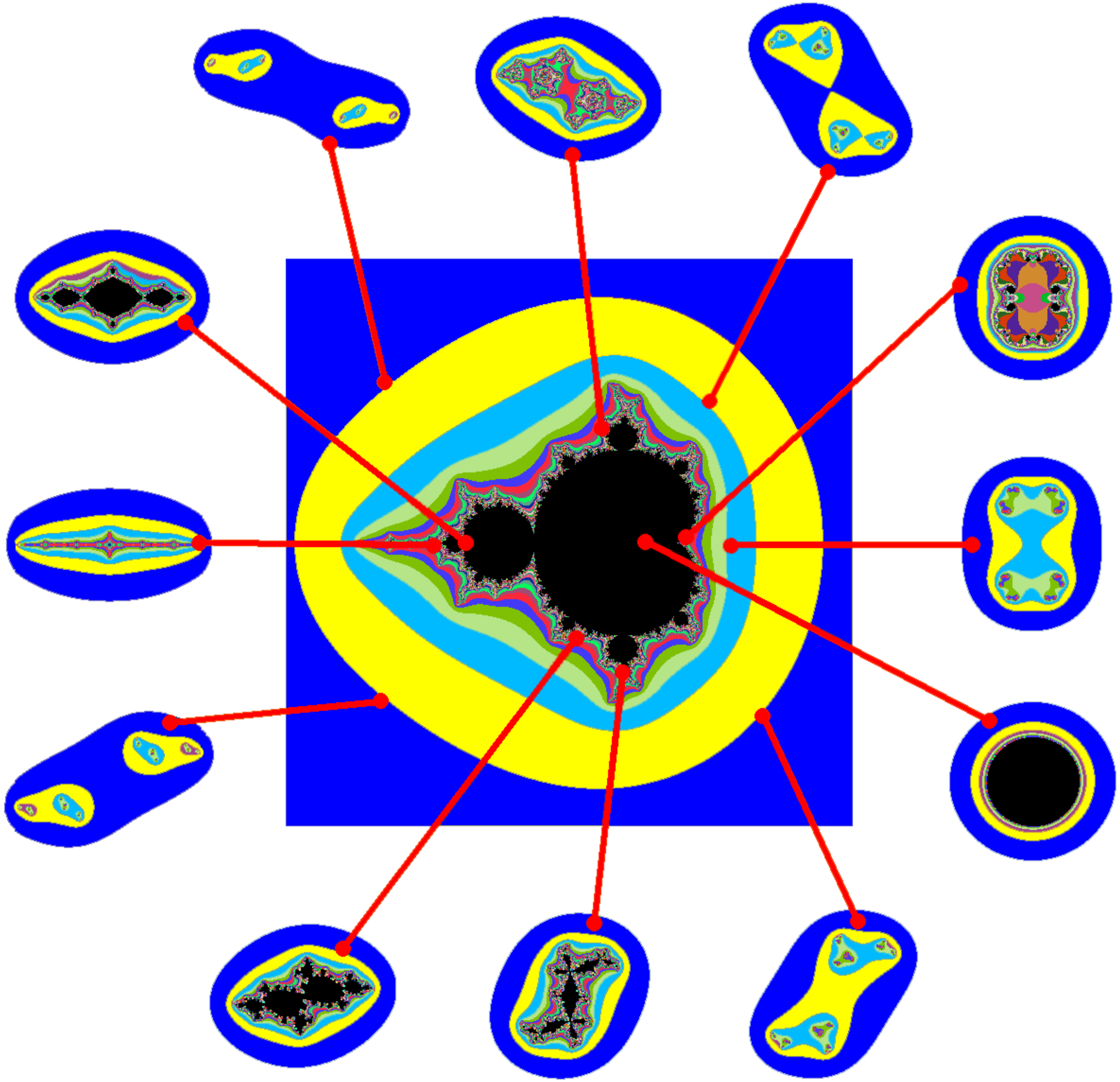

Figure C.1: A much longer caption that will not be listed in the list of figures page.

Appendix D

Presentation Slides

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

Appendix E

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.