Arbitrage Trading Software

Shane Clarke

G00303735

B.Sc.(Hons) in Software Development

April 24, 2016

Final Year Project

Advised by: Damien Costello

Department of Computer Science and Applied Physics
Galway-Mayo Institute of Technology (GMIT)

Contents

About this project

Abstract This project is an arbitrage betting web application. [1]In economics and finance, arbitrage is the practice of taking advantage of a price difference between two or more markets: striking a combination of matching deals that capitalize upon the imbalance, the profit being the difference between the market prices. [2]Arbitrage betting is the only way to make constant profits from sports betting. You don't have to hold an interest in sports or betting to take advantage of an arbitrage bet in order to make extra money online. Arbitrage is a trading technique that uses the difference in the price (odds) to make instant profit. I carried out research on arbitrage betting to see first hand the procedure that was used and continued on to look for the algorithms that make it work. Firstly, I tested out the algorithms on paper to see if the calculations were correct, and they were successful. I continued on to build a website scraper to gather information from the betting websites, which I could use in the algorithms. I carried out tests to verify it was working correctly. I recieved funding from a number of people and we carried out tests in order to make sure that it worked. We turned €3000 into approximately €4500 in a two week period. From there, I went on to add more sports and websites so that the system could find more opportunites. After I had five bookmakers websites, I proceeded to build my own website so that people could create an account and view the results of the system and place their own bets.

Authors Shane Clarke - Galway Mayo Institute of Technology

Chapter 1

Introduction

Arbitrage betting I will begin with definition of arbitrage betting. Arbitrage betting is a win win when it comes to placing bets, if it is done correctly and with patience. You have to find a match where the odds of each contender are above a certain price. The only way to really do this is by using bookmakers against each other.

Bookmakers systems are so complex and well built that they rarely lose more money then they take in. As a result of this, you have to search each bookmaker and locate the same match and compare the odds that are displayed to you. Lets take a tennis match for example, if bookmaker 1 gives you odds of 2.3 on player 1 and 1.6 on player 2, that would not be an arbitrage bet because the odds given on the two players are not high enough. Then you go to bookmaker 2 and they give you 2.0 on player 1 and 2.0 on player 2, again it would not be an arbitrage bet, but if you bet on player 1 on website 1 and bet on player 2 on website 2 you are guaranteed to make a profit. This is what arbitrage betting is, finding a match where the odds are high enough that you can cover all outcomes and ensure that you will make a profit.

The website I used to do all my research on arbitrage betting was [2] http://www.sportsbettingworm.com. It contains alot of information on what needs to be done in order to carry out arbitrage betting successfully. This website provided me with the algorithm needed to find an arb(arbitrage) bet and gave me advice on placing the bet and some good bookmakers to do it with.

To prove to myself that the algorithm worked correctly I visited several bookmakers websites and started looking at matches. I filled in the values into the algorithm. Everytime the algorithim provided me with a correct result not specifiically an arbitrage bet, but a correct result of the algorithm.

Website scraper After testing the algorithm I was sure it worked well and started thinking about how quick the values changed on the bookmakers websites. It was then that I started building the website scraper to gather all the odds of tennis matches from different bookmakers.

I was building the scraper in java and had to find a library that could carry out the right job for my needs. I found the [3]jsoup website from which I downloaded the jsoup java html parser. This java library allowed me to scrape bookmakers sites where the odds were displayed by html.

I spent a while learning how the jsoup library worked and then began scraping a page from one a bookmakers site that displayed all the tennis matches for the present day. Unfortunatly, jsoup scraped the enitre web page and I had to constantly test that I was just scraping the correct infomation. It took a while to weed out the parts of the page that I did not need.

After I got the first webpage parsed I visited a second bookmakers website and located all the tennis matches for the day and scraped that page aswell. I then had enough data to be able to use the algorithm in my scraper. Again I had to figure out how to implement the algorithm into my scraper. After I got the algorithm implemented correctly I started seeing alot of opportunities where money could be made. I then placed bets based on the results of my application.

Testing the scraper It was friends of mine that established the idea, while I carried out the research to make it a reality. For the first week we deposited €10 into each account and made small bets to see whether or not it would work. After the first week we made over 100% profit on the money we had invested. After seeing the results from the first week, we recieved around €3000 euro investment and then the fun started.

I started leaving the system running 24/7 from my own home and it started finding arbitrage oppertunities at all times of the day. The only downside was that most of the arb chances it found were on live matches because the odds are changing so quickly. This meant that because they were live matches there was more risk involved. We set up shop in my friends home because we needed a different ip address than the one that was scraping the odds. The reason for this is because bookmakers are all about making profit and do not like when you find a way around their money making schemes. From there we started investeing big money based on the results of the application. Within two weeks we made nearly €1500 profit on the initial investment. The bookmakers started getting suspicious and limited our accounts so we could only place small bets of only €1 - €2 on the matches which would result in minuscule returns and so we finished up

at that and divided up the profits. I continuied to make it better.

Building the website When we finished the testing phase of the scraper I kept on improving it and slowly added a few more bookmakers and more sports including snooker and darts. I then began building the website that would allow other people access to sign up and receive the results produced from my scraper application. I was going to build a desktop application but it was suggested to me that I should build a web application because desktop applications are dying out, A web applicatio also makes it alot easier for people to sign up and to view on the web. As most people in todays society own smart phones or tablets that would also allow them to view it on the go.

I used jsp (java server pages) to build the website, as it was only a basic site needed to display the scrapers results to the webpage and also I wanted to learn the jsp technology. I had no knowledge of the jsp technology so I had to learn it from scratch. I found a very good tutorial on [4]udemy website. Udemy is a online course website that provides many online courses, quite a few on computer programming and the likes. So I bought the [5]course and started learning jsp. It was a very in depth and well explained course. When I had completed the course I started building the site and it was not long before the site was up and running.

I built the front end to the site and then thought I would need to store all my database results somewhere better then on my localhost database, on my own pc. So I bought some hosting space on [6]GoDaddy the domain registrar and web hosting company to store all the database results safely and so I could access them on any computer.

Chapter 2

Context

Chapter descriptions

- **Chapter 1:** Provides a brief description of each part of the project that I undertook.
- **Chapter 2:** Gives information of the objectives and a link to the code on Github.
- **Chapter 3:** Explains the business methodology and the software methodology.
- **Chapter 4:** Reviews the technologies used throughout my project.
- **Chapter 5:** Explains the design of the system, the architecture, UML diagrams and screenshots.
- **Chapter 6:** Evaluates the systems preformance and robustness.
- **Chapter 7:** Conclusion of the project. How it turned out.

Objectives

- **Why I choose arbitrage trading**

  I have always had a little intrest in gambling as I am a keen sports fan. But as everybody knows the famous saying "The house always wins". I had the idea that because all the major bookmakers are online I would be able to find opportunites where they leave themselves open to arbitrage bets similar to wall street. In brief I hate to lose and the same as everybody else I hate to be at a loss of money.

- **Algorithm implementation**

  Therefore my objectives were to learn how to implement the algorithm I found on [2]sportsbettingworm.

- **Build a scraper**

  Successfully build a scraper which could be used to scrape bookmaker websites but also news websites in which I could use for future projects as I feel data is a very valuable asset.

- **Make money**

  I wanted to create a project that had real life value and from which I could get paid for my work instantly. Then build it in a fashion which would allow it to be expanded on to make even more money.

- **Provide opportunities**

  Provide other people with the opportunity to beat the bookmakers by building a website to display the odds on it and in future sell advertising space and charge people for access to this data.

- **Business aspects**

  Learn the business aspects of the gambling industry so I could compare them to wall street or financial trading in future as I have a keen intrest in entrepneurship.

2.1 The combination of my objectives

Once I knew what I had to do, my objectives were clear. I knew I was creating a project that I had interest in as I find all of the above objectives very interesting and challeging in ways.

I could then see the value of my four years of study in GMIT come to light. I was creating a product which I knew I could make a profit from, whilst giving normal people a opportunity to beat some of the gambling industries biggest corporations. I also knew that the gambling industy has some of the most complex websites out there. Both mathematically and from an end users expirence point of view.

2.2 Technology entrepenurship

When I researched what subjects I would study in college, I leaned towards software development. When I researched software development I found

news articles about founders of technology companies. People such as Mark Zuckerberg, Larry Page and Sergey Brin and these people like me have the ability to build software but the one aspect which they had to learn, which I am learning through this project is business intellect.

These people have created companies which have changed peoples live for the better on a global scale. Also these people have created enterprises worth billions and billions of dollars at a quicker rate than any other business person in living history. This combination of building software and having a business aspect to it cemented in my mind that this would be a great project for me to learn from.

2.3 Github Repository

Github address:

https://github.com/shaneclarke93/ArbSoftware.git

Github contains all my source code, for all the parts of my project, the scraper application and the website. It also contains a copy of the database I used in the development and contains a short video of my application working and instruction in using my project. I have also put the excel spreadsheet of the investment we received and the profit we made while testing the software and a UML diagram of the arbitrage scraper application.

Chapter 3

Methodology

3.1 Business methodology

My business methodology was to take as much advice from local tech entrepreneurs as possible, so I would be efficient with my time and learn from their mistakes in business.They could give me the quick corridor and show me the shreud ways of making money online without losing, as I know from research that [8]8 out of 10 startup companies fail.

I also got their advice on how to raise funding for a piece of online technology and how to give the users experience more of an impact. Both of my ideas to talk with experienced entrepreneurs payed off tenfold as they showed me alot of pitfalls to avoid.

3.2 Software methodology

I took this project on a monthly basis and gave it good time, as well as balancing all my other college work. I had regular meetings with my friends who originally formed the idea for this application. We discussed how things should be done in depth and as we were discussing them I was putting them into action while coding the project.

I choose java as my language for building the scraper as it suited my needs, as I saw fit. Also it was the main language I used throughout my four years in college. I had planned to use javafx to build a desktop application but was advised that I would be better off doing a website for it as desktop applications are dying out. I found this the perfect opportunity to learn a new technology and so I learned java server pages and built the website using

that.

In order for this idea to work I had to implement an algorithm I found on a [2]website containing information about arbitrage betting. As this is the only algorithm I know of, that can find arbitrage trades I felt it was the best algorithm to use and it worked perfectly as planned.

One of the troubles I ran into was when I was scraping different bookmakers websites, was that they displayed the names of people and teams differently. The names were pretty close but not the exact same so I had to find an algorithm that could compare strings and return me a percentage of how closely the strings were matched. I did quite alot of research for this online and found the Levenshtein distance algorithm suited my needs perfectly [9]rosettacode had a very good example of this method. I thought it was the correct algorithm for the job as the strings I would be comparing are not to large and because of that, the time complexity would not be bad enough to affect the systems performance.

As I only need to store basic small tables of data I used MySQL for my databases as I know that database technology the best and it suited my needs nicely.

When I had built the website scraper and implemented the algorithms I found online, I tested that part of the project in the real world by getting investments and making real profit from that.

When I had completed building the scraper I went onto developing the website so users could get the benefits of arbitrage trading and win some of their money back. This is when I learned jsp for building my website.

Chapter 4

Technology Review

4.1 Java

Java was selected as the programming language for the arbitrage scraper because I have been studying it over the last 4 years in college and I am experienced in it. Some other reasons I choose java are because:

- It is an object oriented language - As I am scraping multiple websites I would be doing part of the same thing over and over again and it would be redundant having to have the same code in all classes that scrape a different website.
- Great free integrated development environment - Eclipse is a great IDE for developing java applications and its free. I have being using eclipse for 3 years and know alot of the features it offers.
- It is a cross platform programming language so when I have the application built it can run on any OS such as OS X, Windows or Linux.
- Java has automatic garbage collection so there is no need to worry about manually collecting objects when they go out of scope.
- It has got a vast array of 3rd party librarys to choose from. One of which I found was Jsoup the html parsing library (For info on Jsoup see below).
- Javs's multithreading - I used java's multithreading capabilities when I was doing the scraping of the webpages. I have every page that needs to be scraped in its own thread so that the program will run faster. I needed the program to be fast because arbitrage bets can change in seconds.

4.2 Jsoup

[3]Jsoup is a java html parser. It is a library that is used to extract and manipulate data from a url or html file. I choose jsoup as the library that allowed me to scrape websites because I thought it suited my needs perfectly.

Jsoup uses CSS, DOM and Jquery like methods to extract and manipulate data from webpages. Jsoup can be tedious in the fact that if your trying to get some specific data from a webpage you have to try a few different ways of getting it and it can still give you back the wrong data. The selector tags you use have to be specific to get the correct data.

4.3 MySQL

MySQL is a open source relational database management system that uses structured query language for adding, accessing and managing the content in a database.

The reason I choose it for my project is because I am experienced in it and I only needed a simple database to store the users of my website and the results of my arbitrage scraper.

Java's JDBC (java database connectivity) is an API for java and it works really well with MySQL databases. Also I have used JDBC with MySQL before and I knew that it would be simple to implement and use.

It is also easily scalable and flexible, so that if in the future I want to upscale my project it would not be hard and I could make it as large or keep it as small as my needs see fit.

4.4 HTML, CSS, Javascript

HTML, CSS and Javascript are all front end web technologies. These technologies provide a user something that they can actually see and interact with, without these there would be no web for basic computer users to use.

I needed these web technologies in order to build my website so I could display the result of my arbitrage scraper to users all over the world. I choose to do a website and not a desktop application because I was advised desktop

apps are dying out now, plus most people now of days own a smart phone or tablet. I would reach a much further area in the market then just providing it to people with a laptop or desktop.

4.4.1 HTML

HTML stands for hyper text mark up language and is the most basic thing needed for a webpage. A html page of a website contains all the headings, paragraphs, titles, images and pretty much just all the content for the page to display.

Html uses tags to display the different types of content on the page. Html tags normally come in pairs with an opening tag and a closing tag and the content goes in between these tags.

Html works with css to make the web page user friendly and not just blocks of text around each other.

4.4.2 CSS

CSS stands for cascading style sheets. These style sheets are used to style html pages and give a user friendly look to website. They use classes and ids of tags in a html page to style different things that are on the webpage. They can change the position, size, color, background, font and many other characteristics of text, images and links on a html page.

Style sheets are very convenient because if you have a website with hundreds of html pages and wish to change one characteristic on all of the pages that contain that characteristic, providing they all have the same class, you just have to go to the style sheet that contains the class and change the one or two lines in that style sheet. That will change the same item on all pages throughout the website. In short it reduces redundant code.

4.4.3 Javascript

Javascript is a high level, dynamic programming language. It is the programming language of the web. Alongside html and css it is one of the three core technologies of the world wide web. Javascript is used to do things dynamically on webpages. To allow tasks on a webpage to be completed without having to make a request to a server and load a new page.

An example of javascript being used would be when someone is filling out a form and they put in an invalid password, javascript would alert the user that it is invalid without the page having to be submitted to the server to see if it was valid or not.

4.5 JavaServer Pages

[X]JavaServer Pages are used to create dynamically generated webpages based on html and other document types. JSP is similar to PHP and ASP but it uses the java programming language. In order to deploy and run javaserver pages, a compatible web server is needed such as apache tomcat (for more information on apache tomcat see below).

I choose to use javaserver pages to build my website because I wanted to learn a new technology and already know java so I assumed it would not be to hard to pick up.

Javaserver pages can use the REST(Repersentational state transfer) api style architecture in the development of websites, instead of having to use session objects to store ids and passwords, they can be passed from page to page with a request.

4.6 Apache Tomcat

[11]Apache Tomcat is an application server that executes java servlets and renders web pages that include jsp coding. Because I was developing my website using javaserver pages I needed something that I could deploy and run my website on and the tomcat server was the perfect solution for this.

Some components of the apache tomcat server are:

- Catalina is tomcats servlet container. It implements sun microsystems specifications for servlet and javaserver pages.
- Coyote is a connector component for tomcat that supports HTTP 1.1 protocol as a webserver. Which allows catalina to act as a plain web server.

- Jasper is tomcat's JSP engine. It parses JSP files to compile them into java code as servlets(that can be handled by catalina). At runtime jasper detects changes to jsp files and recompiles them.

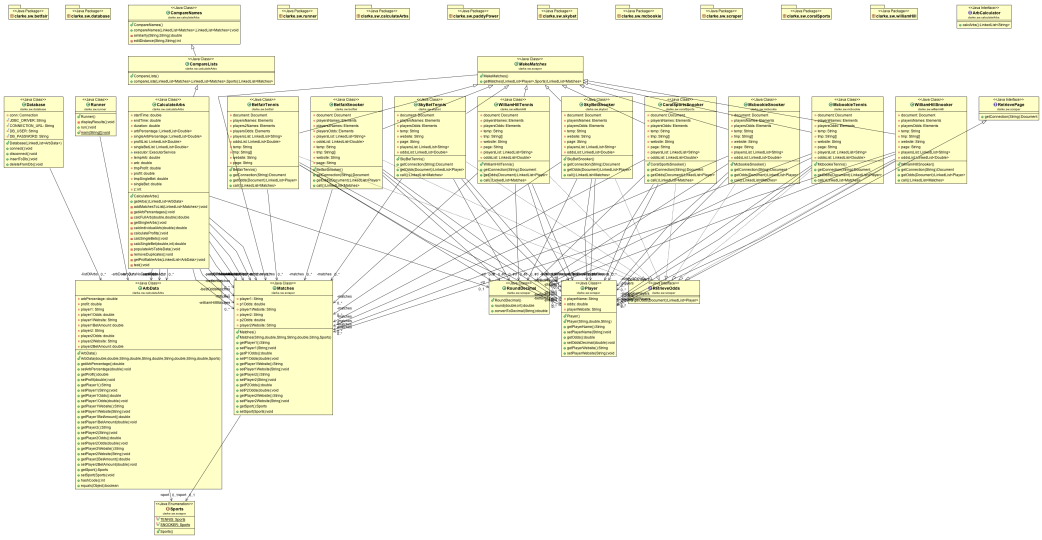
Chapter 5

System Design

I designed the system by building two seperate applications. I have them sharing the same database where one application provides the database with the data and the other application displays the data to the user.

The first application I built was the arbitrage scraper, there is a small UML diagram below. Its not very easy to read so the full size image is on my github repository: https://github.com/shaneclarke93/ArbSoftware.git

Arbitrage scraper UML diagram



Alot of research was put into building the webscraper. I first had to find a good library in java that could scrape websites, also I had to research bookmakers websites and locate a few where the odds were displayed by html because that is what the java library I found could scrape.

When I found a few websites and the correct library to use, I had to go through each page that would be scraped individually because I had to parse only the data I wanted from them.

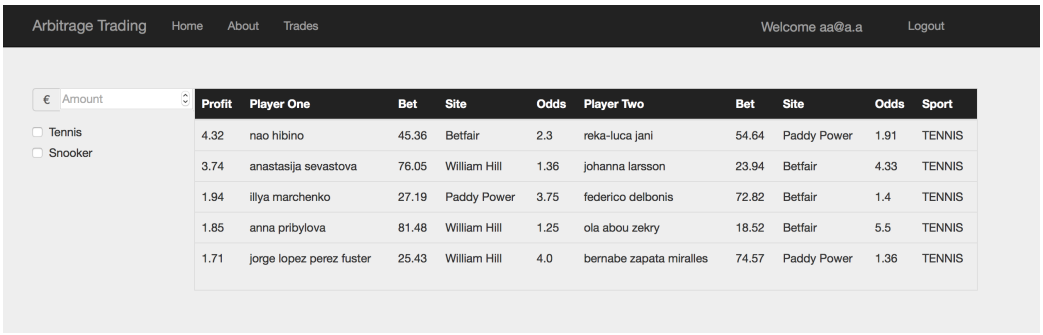
After I had the data I needed from the webpages, I had to think about how to implement the algorithm to find the arbitrage bets. This was not that simple as there are a few different calculations that are used in the algorithm.

Once I had at least two webpages scraped and the algorithm implemented I could build the website that would be able to display the odds to the users.

In order to do this I had to learn the technology that I was going to build the website with, which was jsp. I found a great tutorial on java server pages and set to work on learning it. Once I had finished the course I set to work building the website which uses jsp as the server side code and html, css, bootstrap, javascript and jquery as the client side code.

I had to decide what would be the best way for the user to view the bets. So I decided to go with a table that showed all the information needed in order to place an arbitrage bet. See screenshot below.

Arbitrage Information Table



Profit	Player One	Bet	Site	Odds	Player Two	Bet	Site	Odds	Sport
4.32	nao hibino	45.36	Betfair	2.3	reka-luca jani	54.64	Paddy Power	1.91	TENNIS
3.74	anastasija sevastova	76.05	William Hill	1.36	johanna larsson	23.94	Betfair	4.33	TENNIS
1.94	illya marchenko	27.19	Paddy Power	3.75	federico delbonis	72.82	Betfair	1.4	TENNIS
1.85	anna pribylova	81.48	William Hill	1.25	ola abou zekry	18.52	Betfair	5.5	TENNIS
1.71	jorge lopez perez fuster	25.43	William Hill	4.0	bernabe zapata miralles	74.57	Paddy Power	1.36	TENNIS

I have the default bet amount set to €100 but I have put an input box to the left of the table so the user can set there own bet amount in case they dont want to place €100 on each arbitrage bet.

Chapter 6

System Evaluation

6.1 Robustness

6.1.1 Robustness in usage

My software is robust because as you can see from day one on the chart below, that when I placed one of my first trades I made a profit and if I was to place another one today it would still produce a profit.

[7]Excel file screenshot could not fit here so refer to accounts file in github repository for the figures.

The outcomes can be measured by the amount of profit we made from a minimum investment of both time to create the piece of software and the amount of money returned to us. Also another measurement of my success was that I recieved alot of positive endorsements from the local entrepreneurs that I asked for advice on my project. They informed me that they would offer me a job at anytime in their company or introduce me to other companies which I wanted to work at.

One of the biggest limitations for me was I could only put a certain amount of time into this project as I have other college responsibilities. The one major limitation in this project was that after a few weeks of trading and making a nice profit the bookmakers limited the amount of money we could place on a bet thus limiting the amount of profit we could make from each bet.

However the huge opportunity is to enter the bookmakers on a new account and place the biggest possible trade you can, for example a €10000 trade with a %60 profit margin would give you a €6000 profit instantly for

two clicks of a button.

Another opportunity for me is to use this technology to compare the global financial markets which is similar to the technology used on wall street. Although basic compared to theirs I still have the fundamentals correct and with some training and experience I believe I could be the next wolf of wall street in terms of technology used.

6.1.2 Robustness in running

My software is robust because I have exception handling built in to it for instance, I have it scraping over 10 pages but all them pages may not have matches going on at the time of scraping so they will return no results and if that happens I have it skipping them pages when it comes to the comparsion of the matches on each website. So the program wont just crash with a null pointer exception.

6.2 Performance benchmarks

I need my application to be fast in order to find an arbitrage opportunity you may only have 30 seconds to get the data and place the bet. So while I was building the application I had to keep checking the speed of the scraping of the pages, the comparsion of the names of players/teams and the speed of comparing the odds.

When I had began the project the speed of scraping the pages was horrible so I had to think of a way around this. I decided to use multithreading in order to scrape all the pages I needed at the same time and because I was using java I decided to use an executor service so I could give each thread 10, 20 or X amount of seconds to run and return me a page and whether they had the page or not I could carry out the algorithm on the pages that it did scrape.

After I had it scraping the pages in a decent time I had to go about finding a way to compare the names and I found the [9]Levenshtein distance that compares strings and returns a % out of 100 of how close the strings are to each other. This worked fine because all the strings I had to compare were very short most were less then 20 characters and all were less then 30 characters so this runs pretty quick in well under 1 second.

6.3 Outcomes vs objectives

I think the outcomes of the project stack up very well against the objectives of the project. One of my objectives was algorithm implementation and the way I used the algorithm for finding arbitrage opportunities was executed perfectly and it had to be, otherwise the project would have failed and I would have lost a large sum of money and wasted alot of time building this.

Building the website scraper was another objective of this project and I think I did this correctly because my scraper does exactly what it is meant to do and get just the information I needed in order to input it into the algorithm I found.

Making money was one of my favourite objectives and I can gladly say that myself and my friends accomplished this objective. In making nearly €1500 profit in just two weeks.

Chapter 7

Conclusion

With my project I now have the ability to discover a business opportunity and make real financial gain but more importantly from a tech point of view I know from all the obstacles I encountered while building this project I have the ability to solve problems no matter how hard they seem at first. I also found that I have a skill to communicate in order to motivate people to help me solve a problem which I may not be able to solve on my own.

My conclusion is that my objectives were brilliant for the industry that I choose to do this project on. I may have been lucky in the objectives I choose, but I know now that by choosing these objectives I saved myself precious time by having objectives as a guide. By having the objective of making money I aslo impressed techonlogy entrepeneurs who now see me in a different light. They now know that I have learned alot of new business and financial ways of thinking.

In the final conclusion of my project as a whole, if I was to do an arbitrage trading project again I would start by doing it on a different industy. The industry I believe I would use is comodoties trading, stock trading or financial currerency trading as the regulations on these industries allow you as a trader to legally place arbitrage trades from an automated computer service unlike the gambling industry which has to be done manually by a human. Which is both time and energy sapping not to mention financially expensive. I also know that the move in technology in the financial capitals of the world is always looking to hire graduates who have a tech background in financial trading. Therefore I would get a job doing two things I enjoy creating software and making money. In a place in the world which would have a great lifestyle.

I was very happy with the end result of all the aspects of this project both in terms of learning different types of software and implementing the different types of software.

Even though my project gives you the best possible chance of beating the bookmakers I will leave on this cautionary note that bookmakers will do everything possible in order for them to win.

Bibliography

[1] Wikipedia Arbitrage,
https://en.wikipedia.org/wiki/Arbitrage

[2] Sports Betting Worm,
http://www.sportsbettingworm.com/arbitrage-betting/

[3] Jsoup,
http://jsoup.org

[4] Udemy,
https://www.udemy.com/courses/

[5] JavaServer Pages Course,
https://www.udemy.com/javawebtut/learn/v4/overview

[6] GoDaddy Website,
https://ie.godaddy.com

[7] Github Address,
https://github.com/shaneclarke93/ArbSoftware.git

[8] Forbes Startup Fails,
http://www.forbes.com/sites/ericwagner/2013/09/12/five-reasons-8-out-of-10-businesses-fail/#70f3d2eb5e3c

[9] Rosetta Code Levenshtein Distance,
https://rosettacode.org/wiki/Levenshtein_distance

[10] JavaServer Pages,
https://en.wikipedia.org/wiki/JavaServer_Pages

[11] Apache Tomcat,
https://en.wikipedia.org/wiki/Apache_Tomcat