

CO3014 Computer Science Project

Final Report

Virtual Stock Trading

Student: María Mercedes García Martínez

Supervisor: Dr. Stanley P. Y. Fung

*Project Report submitted to the University of Leicester in
Partial Fulfillment for the degree of Bachelor of Science*

May 2009

Department of Computer Science,
University of Leicester

This project could not have been realized without the help of some persons. Firstly, I am grateful to my supervisor Stanley Fung to guide me in the project and the University of Leicester to give me the chance to do this work.

Moreover, I am also grateful to my family and friends to help me to decide to make the project.

Table of Contents

<u>1 Introduction.....</u>	<u>1</u>
<u>1.1 Aims.....</u>	<u>1</u>
<u>1.2 Objectives.....</u>	<u>4</u>
<u>1.3 Background Research.....</u>	<u>5</u>
<u>1.3.1 Trade.....</u>	<u>5</u>
<u>1.3.2 Database.....</u>	<u>6</u>
<u>1.3.3 Uniform Server.....</u>	<u>6</u>
<u>1.3.4 PHP.....</u>	<u>6</u>
<u>1.3.5 MySQL.....</u>	<u>6</u>
<u>1.3.6 Apache.....</u>	<u>7</u>
<u>1.3.7 Graphics.....</u>	<u>7</u>
<u>1.3.8 PDF files.....</u>	<u>7</u>
<u>1.3.9 RSS.....</u>	<u>7</u>
<u>1.3.10 CSS.....</u>	<u>7</u>
<u>1.4 Literature Survey.....</u>	<u>8</u>
<u>2 Planning and Timescales.....</u>	<u>9</u>
<u>3 Project Core.....</u>	<u>11</u>
<u>3.1 Technical and Engineering Background Information.....</u>	<u>11</u>
<u>3.1.1 Internet, www and browsers.....</u>	<u>11</u>
<u>3.1.2 JpGraph.....</u>	<u>13</u>
<u>3.1.3 FPDF.....</u>	<u>14</u>
<u>3.1.4 RSS.....</u>	<u>15</u>
<u>3.2 Software Requirements Specifications.....</u>	<u>17</u>
<u>3.2.1 Introduction.....</u>	<u>17</u>
<u>3.2.1.1 Purpose.....</u>	<u>17</u>
<u>3.2.1.2 Project features.....</u>	<u>17</u>
<u>3.2.1.3 Definitions.....</u>	<u>17</u>
<u>3.2.2 System features.....</u>	<u>20</u>
<u>3.2.2.1 Functionalities.....</u>	<u>20</u>
<u>3.2.2.2 Function requirements.....</u>	<u>20</u>
<u>3.3 Design.....</u>	<u>21</u>
<u>3.3.1 Use Case Diagram.....</u>	<u>21</u>
<u>3.3.1 Entity-Relationship Model.....</u>	<u>26</u>
<u>3.3.2 Directories.....</u>	<u>27</u>
<u>3.3.3 Interface.....</u>	<u>29</u>
<u>3.3.4 Example of a Demonstration.....</u>	<u>35</u>
<u>3.4 Implementation details.....</u>	<u>39</u>
<u>3.4.1 Tools.....</u>	<u>39</u>
<u>3.4.2 Code explanation.....</u>	<u>44</u>
<u>3.5 Software Testing and Test Sets.....</u>	<u>64</u>
<u>4 Critical Appraisal.....</u>	<u>65</u>
<u>5 Bibliography and Citations.....</u>	<u>66</u>
<u>5.1 Literary Sources.....</u>	<u>66</u>
<u>5.2 Web Sources.....</u>	<u>67</u>
<u>6 Appendix.....</u>	<u>68</u>
<u>6.1 Database with an example of data.....</u>	<u>68</u>
<u>6.2 Queries to create the Database.....</u>	<u>70</u>

DECLARATION

All sentences or passages quoted in this report, if from other people's work have been specifically acknowledged by clear cross referencing to author, work and page(s). I understand that failure to do so amounts to plagiarism and will be considered as grounds for failure in this module and the degree examination as a whole.

Name: María Mercedes García Martínez



Signed:

Date: 8th May 2009

This project is a web-based software system for virtual stock trading. Users can log in the system and they can buy and sell stocks. The virtual system will permit practise and learn how the trade market works without real money.

Virtual Trade uses the technologies such as PHP, JavaScript, CSS, MySQL, Apache, JGraph, FPDF and RSS. It works in a Uniform Server that includes Apache server, PHP and MySQL.

The system records the history of stock prices and provides graphical information tools to present this data. Users will be able to view their personal information, own stocks, transaction history and their current orders to buy and sell and the prices of the stocks. Users can generate PDF files with an abstract of their information. Moreover, they can access with RSS to the purchases and sales.

The users will be able to see how the transactions change and how the orders of buying and selling match them. If there is a buyer who wants to buy a stock for a certain amount of money and there is a seller who wants to sell this stock, it will be a transaction.

The project has been programmed in PHP using queries in SQL. I practised programming and I researched information about the trade market and the technologies JGraph and FPDF to add utilities to the project.

The project is useful to learn how the trade works. It can be used in Economy lessons.

This report tries to explain how the project works and it also explains the tools and technologies used.

Moreover, this project is totally done with free as freedom software, you do not have to pay for licences to install of the programmes needed.

1 Introduction

1.1 Aims

The aim of this project is the design and implementation of a web-based software system for virtual stock trading. Users can log in the system and they can buy and sell stocks. The virtual system will permit practise and learn how the trade market works without real money. Moreover, the advantage is that you can practise anywhere because it is an online application and it is open 24 hours a day.

The trading prices will be obtained by matching the buying and selling orders, each with their target price, as in the real market.

The system will record the history of stock prices and provide graphical tools to present this data. Users will be able to view their personal information, own stocks, transaction history and their current orders to buy and sell and the prices of the stocks.

The system is web-based application designed using appropriate technologies and will support a relatively large number of users.

The application will show a list of the company's stocks with their current prices and a list of buyers and sellers with the name of the stocks that they can buy or sell with the number of stocks and their prices. Moreover, users will be able to see the orders sorted. The best order in the case of buy is the major price will appear first and the worst, in this case the lower price will be appear at last and the orders to sell, the lower price will appear first and the major price will be at last.

Below is an example of how the system will work. For example, there are 30 stocks of the company *Barclays*, *user1* has 10 stocks, he wants to sell them for each £100 and *user2* has 20 stocks, he wants to sell them for each £150. So the Sellers table will be ordered according to the lowest selling price as shown below. And if *user3* wants to buy 5 stocks of *Barclays* for £50 and *user4* wants to buy 5 stocks of *Barclays* for £100, then the Buyers table will be ordered according to the best buying price as shown in the table.

The application will show this:

Sellers:

Company	User	Price	Volume
Barclays	User1	£100	10
Barclays	User2	£150	20

Buyers:

Company	User	Price	Volume
Barclays	User4	£100	5
Barclays	User3	£50	5

The system will do that *User4* buys 5 stocks and *User1* sells 5 stocks to *User4*.

The application shows the following:

Sellers:

Company	User	Price	Volume
Barclays	User1	£100	5
Barclays	User2	£150	20

Buyers:

Company	User	Price	Volume
Barclays	User3	£50	5

Now, if *User4* want to sell 5 stocks, he will be able to put the price for them and he will appear in the list of sellers.

1.2 Objectives

The main objective of my project is to set an Apache Web Server up and running and a Database server in MySQL. The web pages will be done in HTML, PHP, CSS and JavaScript. These tools will permit users connect to the system through the Internet from anywhere with any Operating System and browser.

The data will be saved in the SQL database and the information about the purchases and sales can be accessed with RSS.

The users can see how the transactions change and how the orders of buying and selling match them. If there is a buyer who wants to buy a stock for a certain amount of money and there is a seller who wants to sell this stock, it will be a transaction.

In this application the current price of the stocks in the trade market will be the price of the last sale.

Users would be able to see the graphics about the history of the stock. The graphics will be done with a PHP's library.

Moreover, users can generate a PDF file with his information, his data and his stocks and can get update notifications using RSS.

1.3 Background Research

My background research was generally based on to know how the trade works and how to use the tools that I need to do the project. I read explanations of the trade in books and web pages. I also read books, web pages and manuals about the required tools, Uniform Server, PHP, MySQL, Apache, JGraph, FPDF, PDFlib, RSS and CSS.

1.3.1 Trade

To do this project I had to try to understand how the trade works. In the following paragraphs I will explain it.

The trade is a market where the people negotiate products. The buyers and sellers are in contact. In the trade there are two important figures: the companies and the buyers and sellers.

The companies that need money to achieve their objectives have different ways to get it, one of the most interesting ways is to sell stocks to the trade market.

On the other hand, the buyers and sellers want to get money and they buy the company's products to sell it later.

The trade does that and the economy grows. That is good for the companies and also for the buyers and sellers.

Moreover, the sale price of the stocks is decided by the seller and buyer.

It is important that this information do these things. All of the details should be known in a fast and clear manner to all participants because they need to have equal opportunities.

There are two possible ways of setting trading prices: by retrieving real stock prices dynamically from standard sources, and using them as trading prices, or by matching the buying and selling orders, each with their target price, as in the real market. In this project it will be using the second way. The buyers and sellers will decide the prices when a stock will be sold.

I found information about trade in the Internet in these web pages: <http://www.invertironline.com/Aprender/Nivel4/N4introbolsa1.asp>[12], <http://money.howstuffworks.com/personal-finance/financial-planning/stock.htm>[13] and <http://uk.finance.yahoo.com/>[14]. Reading the information in these web pages I have understood what is trading and its implications. Also, I collected information from the book called "*Arithmetic out at work, a practical course with examples taken from the retail trade*" wrote of W. R. Page[1] about how to work the trade.

Moreover, I spoke with an economist who explained me how the trading works by referring to some real time trading experiences.

In the web site <http://www.invertironline.com/Aprender/Nivel4/N4introbolsa1.asp>[12], you can find the definition and the function of the trade for people that are not specialist in economy. Reading from how stuff works, <http://money.howstuffworks.com/personal-finance/financial-planning/stock.htm>[13], I understood how trading works and with the help of Yahoo finance, <http://uk.finance.yahoo.com>[14], which has a lot of graphical representation of real trading examples.

The books “*Deciphering the market*” and “*Principle of Chart Reading and Trading Stocks, Commodities and Currencies*[2].” helped me to understand the trading concepts in depth and also explains the rule of trading and how the transactions takes place in Bank of England.

1.3.2 Database

I need a Database to keep the data. I designed a Database in accordance with the objectives. I read the book *Databases Solutions* [3], this book helps designing the database. There are information about the database analysis and design techniques, logical database design and physical database design which I used in creating my database.

1.3.3 Uniform Server

I could do the project using the Uniform Server, you can find it in the web site: <http://www.uniformserver.com/>[15]. Uniform server includes PHP, mySQL and the Apache server together, and it is easy to install. This website helped me to understand how the uniform server works and also helped me in installation.

1.3.4 PHP

I searched functions of PHP from this site <http://php.net/>[16], which helped me in implementing the PHP concepts in my project from the PHP manual given in the website.

1.3.5 MySQL

I read the information about MySQL in the web page: <http://www.mysql.com/>[17] In this web page you can find documentation to know how mySQL woks. I consulted this web page: <http://www.w3schools.com/sql/default.asp>[18] to understand about SQL queries and the way it works.

1.3.6 Apache

I found information about Apache server in the following web site: <http://www.apache.org/>[19], where you can find documentation and explanations of the Apache's foundation.

1.3.7 Graphics

The PHP's library called JGraph helped me with the Graphics. I consulted the web site: <http://www.aditus.nu/jgraph/> [20]. It is easy to add in the project as it is written in PHP. Moreover, I found a manual to know how to use it in this web.

1.3.8 PDF files

I wanted to do my project such that it is capable of creating PDF files which could help the users to save the necessary details easily.

Firstly, I found a PHP's library called PDFlib, from the web site: <http://www.devshed.com/c/a/PHP/PDF-Generation-With-PHP/> [21]. But, I couldn't make it work because Uniform Server did not support PDFlib as Uniform Server is not compiled with PDFlib support.

Finally, I found another PHP's library called FPDF, from the website: <http://www.fpdf.org/>[23]. FPDF is a PHP class written in pure PHP, without using the PDFlib library. F from FPDF means Free: you may use it for anything and modify the code if you want. I read the tutorial included with the library and I could easily create PDF files with FPDF.

1.3.9 RSS

I created RSS using a XML file. I learnt how to create this file from reading this website: <http://www.downes.ca/cgi-bin/page.cgi?post=56> [22]. It explains the labels required to do it.

1.3.10CSS

I used CSS (Cascading Style Sheets) to make my project look more attractive and appealing and also user friendly. I found CSS by studying the book "*Cascading Style Sheets* [4]".

1.4 Literature Survey

I used various books and web references to do my project and my report. Web references were more informative and easy to comprehend than the books, to find out how to perform some functions in the program. I used the books to read about trading and to know features of some used tools and to write the report.

I used the following books and websites to learn more about trading:

- *A practical course with examples taken from the retail trade* [1].
- *Principles of Chart Reading and Trading Stocks, Commodities and Currencies* [2].
- <http://money.howstuffworks.com> [13].
- <http://uk.finance.yahoo.com> [14].
- <http://invertironline.com> [12].

I read the following books and websites to learn how to program for my project:

- *Database Solutions* [3].
- *Cascading Style Sheets* [4].
- <http://www.uniformserver.com/> [15].
- <http://php.net/> [16].
- <http://www.mysql.com> [17].
- <http://www.w3schools.com/sql/default.asp> [18].
- <http://www.apache.org/> [19].
- <http://www.aditus.nu/jpgraph/> [20].
- <http://www.devshed.com/c/a/PHP/PDF-Generation-With-PHP/> [21].
- <http://www.downes.ca/cgi-bin/page.cgi?post=56> [22].
- <http://www.fpdf.org/> [23].

I read part of the following books and WebPages to write the report:

- *Dreamweaver MX* [5].
- *The Definitive Guide* [6].
- *PHP and MySQL Web Development* [7].
- *MySQL* [8].
- *Introductory Concepts and Techniques* [9].
- *Macromedia Dreamweaver MX 2004* [10].
- *Firefox-Thunderbird* [11].
- <http://wikipedia.org> [24].

2 Planning and Timescales

The Planning and Timetable of my project:

- The 1st week (16th February – 22nd February) I did first analysis of the software.
- The 2nd week (23rd February – 1st May) I did the first design of the software.
- The 3rd week (2nd May – 8th May) I started with the implementation and run the servers.
- The 4th week (9th March – 15th March) I did the implementation of the GUI.
- The 5th week (16th March – 22nd March) I did the implementation of the GUI.
- The 6th week (23rd March – 29th Mach) I coded WebPages using PHP and implemented it.
- The 7th week (30th March – 5th April) I coded WebPages using PHP and implemented it.
- The 8th week (6th April – 12th April) I designed the Database for storing data and implemented the GNUplot graphics.
- The 9th week (13th April – 19th April) I did the XML files and the PDF files.
- The 10th week (20th April – 26th April) I tested the Software.
- The 11th week (27th April – 3rd May) I started planning for my report.
- The 12th week (4th May – 8th May) I wrote the report.

I followed this plan, with a few minor changes and this was the work that I did during these period:

- The 1st week: I did the design and analysis, the Software Requirements Specifications and the UML diagrams (class diagram and Use cases).
- The 2nd week: I searched in the net a server with PHP, MySQL and Apache, I installed and run the server, I began to design and put data in the Database and I began to do the design of the main page.
- The 3rd week: I designed the Database and began the implementation, I did the login and logout of the users, I was programming in PHP and JavaScript.
- The 4th week: I was programming functionalities and continuing with the implementation in PHP, I also did the GUI and the code at the same time. I was programming each page.
- The 5th week: I was programming the selling and buying in PHP and JavaScript to show to the user the errors. If for example he can't buy because they don't have enough money, I did queries in SQL and tried to solve errors of the functions.
- The 6th week: I programmed, solved errors and put more data in the Data Base.
- The 7th week: I programmed and solved some more errors and conducted tests to prove that it all works well.
- The 8th week: I programmed, did the tables of the stocks with the prices, showing the variations with the old data.
- The 9th week: I finished the selling, buying, details of stocks, registration of the users and the page for the user can to edit their personal information.

- The 10th week: I added more functionality and I generated graphics, PDFs and RSS. I tested the project.
- The 11th week: I began to write the report.
- The 12th week: I finished the final draft of the report. I also did the last test on the project.
- The 13th week: The deadline of the project was moved to the next week. In this week I did more test and I corrected details of the report.

My timescales were effective and realistic. I thought that I needed more time with the Analysis and Design, as well as with the implementation of the GUI but I used up this time to program. The plan was useful to begin with and to structure the ideas.

I modified the implementation details of the plan because I realized that there is a graphic library and it's better to use this for this project than GNUplot.

I also wrote in the plan that the RSS was about the information of the users but I realized that it is better to do RSS about the new purchases and sales.

3 Project Core

3.1 Technical and Engineering Background Information

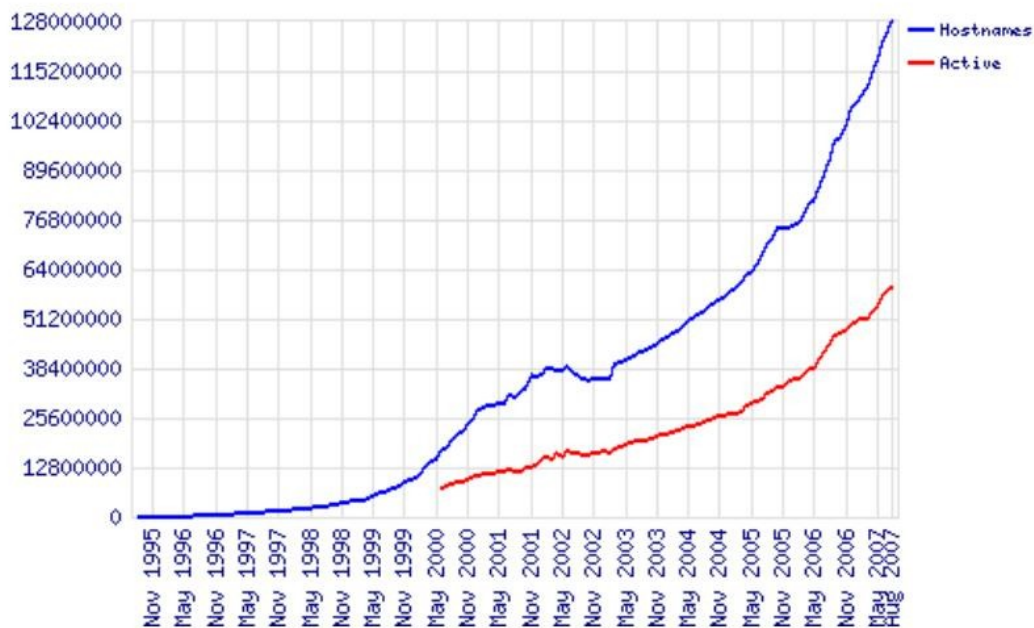
Virtual Trading is a project based on web technologies. I needed to search information about some technologies to do this project. All the technologies used are free, that means that you don't have to buy licenses and you can modify the source code.

It was the first time that I did RSS and I used JpGraph, FPDF. I will explain at follows about Internet, www and browsers and after about RSS, JpGraph and FPDF.

3.1.1 Internet, www and browsers

“The Internet is the most popular and fastest growing area in computing today. Using Internet, you can do research, participate in online auctions, shop for services and merchandise, post a resume and search for a job, buy and sell stocks, check weather and traffic conditions, obtain medical advice, purchase movie and concert tickets, play online video games, listen to online radio stations, and converse with people worldwide.” Firefox. Introductory Concepts and Techniques [9].

The Internet has become an indispensable tool in a few years for many people. In addition, it is the technology that has grown fastest in history. Below you can see an image showing the growth of the Internet.



But you need a good browser to see all the information in the network. Now, there are a lot of browsers. There are two more important browsers: Internet Explorer and Mozilla Firefox.

Internet Explorer was created a long time ago and it is known in the entire world. The statistics [Montenegro, 2006] tell that Internet Explorer is used to 70%-80% of the people that usually use the Internet and Firefox is used to 10%-15% people that use the Internet, this is not important number but is the first time that Explorer has rivals.

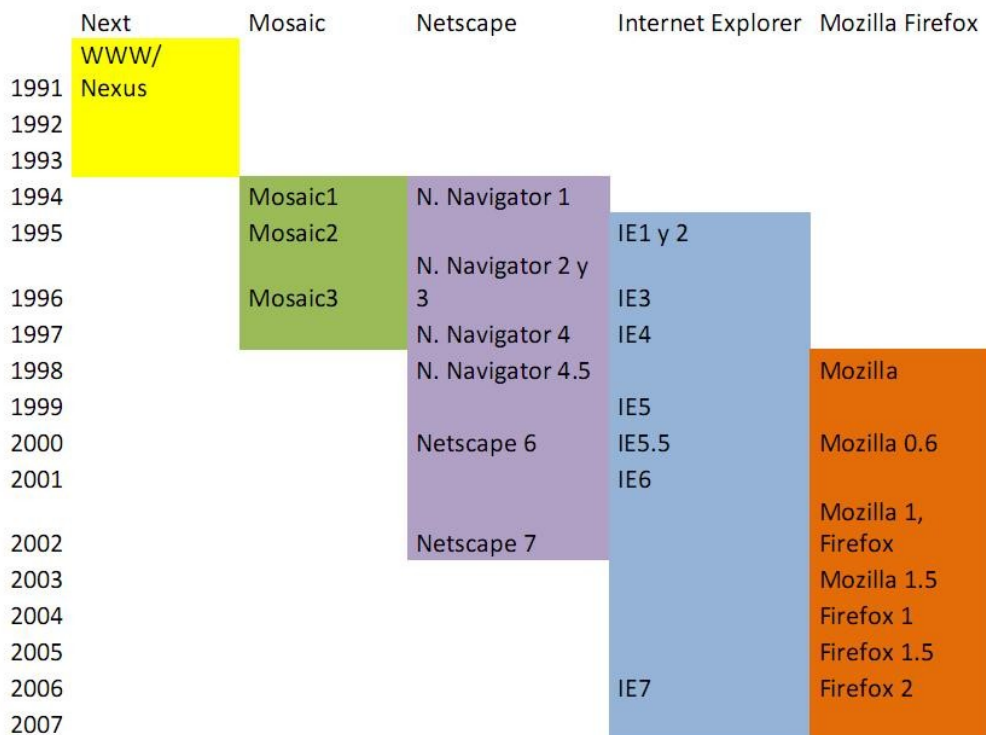
The Internet doesn't have to have rivals it has to be approachable in the entire world. *Firefox-Thunderbird* [11].

Browsers

The first web browser was developed in the CERN in the end of 90's years. It was very sophisticated and it had graphic interface but it only worked in NeXT. The browser Mosaic was the first important browser because NCSA1 prepared versions for Windows and Macintosh. But later Netscape Navigator was created which is much better than Mosaic because it is faster and it had more functionalities.

Internet Explorer was created by Microsoft and it won Netscape Navigator and currently it is the most widely used browser.

In the following image shows the evolution of the browsers:



3.1.2 JpGraph

JpGraph

I used JpGraph version r1096 because it is easy to install with PHP and it is also easy to use. JpGraph is free software. While I was searching information about JpGraph, I found some features that are very interesting in context of this project. In the following paragraphs I am going to describe the library and comment how to use it to draw the graphics in this project.

JpGraph is an Object-Oriented Graph creating library for PHP 4.3.1 and next versions. The library is written in PHP and ready to be used in any PHP scripts (both CGI/APXS/CLI versions of PHP are supported).

It is possible to create numerous types of graphs with this library. JpGraph is easy to use, you can do simple and complex graphs with a few code.

The JpGraph's features are:

- Flexible scales, supports text-lin, text-log, lin-lin, lin-log, log-lin and log-log and integer scales.
- Supports both PNG, GIF and JPG graphic formats.
- Supports caching of generated graphs to lessen burden of a HTTP server.
- Supports batch mode to only generate images to a file.
- Supports client side image maps which make it easy to produce drill down images.
- Auto-scaling which gravitates towards esthetical values.
- Fully supports manual scaling, with fine grain control of position of ticks.
- Supports background images with different formatting options
- Supports colour and brightness adjustments of images directly in PHP.
- User specified grace for auto-scaling
- Supports up to two different y-scale, it is possible to have different left and right y-scale and add plots to both
- Supports, line-plots, filled line-plots, accumulated line-plots, bar plots, accumulated bar plots, grouped bar plots, error plots, line error plots, scatter plots, gantt-charts, radar plots, 2D and 3D pie charts.
- Supports unlimited number of plots in each graph, makes it easy to compose complex graph which consists of several plot types
- User specified position of axis
- Supports colour gradient fill in seven styles
- Designed as a flexible OO framework which makes it easy to add new types of plots
- Supports automatic legend generation
- Supports both vertical and horizontal grids
- Supports anti-aliasing of lines
- Supports background images as well as unlimited number of icons in the graph
- Supports rotation of linear graphs
- More then 400 named colours.
- Designed modularly - you don't have to include code which isn't used

The scripts in JpGraph generates an image that must be in a separate file which is called in a HTML tag. In this project we use the script “graphic.php” that generates in PHP an image and we call this in this way:

```

```

The most important commands to write a graph are:

```
//to create a new Graph  
$graph = new Graph($width, $height, ...);
```

```
//to generate a new Graph  
$graph->Stroke();
```

This information is from: <http://www.aditus.nu/jpgraph/>[20].

3.1.3 FPDF

I used FPDF version 16 in my project to generate PDF files. FPDF is a class written in PHP to generate PDF documents directly from PHP without using the library PDFLib. The mean of F in FPDF is Free, you can use it for any purpose and modify to your liking to fit your needs.

- Choice of measure unit, page format and margins
- Management of headers and footers
- Automatic Page Break
- Line breaks and justification of automatic text
- Admission of images (JPEG, GIF and PNG)
- Colours
- Links
- Admission of TrueType fonts, Type1 and encoding
- Compression of page

FPDF requires no extensions (except zlib to activate compression and GD for GIF support) and works with PHP4 and PHP5.

I needed to create tables in the PDF to show the personal information of the user and the user's stocks.

To create a table you need to write:

```
function BasicTable($header,$data)
{
    //Header
    foreach($header as $col)
        $this->Cell(40,7,$col,1);
    $this->Ln();
    //Data
    foreach($data as $row)
    {
        foreach($row as $col)
            $this->Cell(40,6,$col,1);
        $this->Ln();
    }
}

```

```
$pdf=new PDF();
$pdf->SetFont('Arial','',14);
$pdf->AddPage();
$pdf->BasicTable($header,$data);
$pdf->Output();

```

This information is from: <http://www.fpdf.org> [23].

3.1.4 RSS

I also used *RSS* (Really Simple Syndication). RSS is a family of formats for web sources encoded in XML. It is used to provide subscribers with updated information frequently. The format makes it possible to distribute content without a browser, using software designed to read RSS feeds (usually called aggregator). Despite this, it is possible to use the same browser to view RSS feeds. The latest versions of major browsers can read RSS feeds without additional software. RSS is part of the family of XML formats developed specifically for all types of sites that are updated regularly and through whom you can share information and use it on other sites or programs. This is known as re-organize or web site (a mistranslation, but widely used). <http://wikipedia.org> [24].

I found the information about how to create it showing other RSS files.

A typical file to do RSS is:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<rss version="0.91">

    <channel>
    <title>Title</title>
    <link>http://webpage.html</link>
    <description>Description</description>
    <language>en-uk</language>

    <image>
    <title>Title</title>
    <url>http://image.gif</url>
    <link>http://webpage.uk</link>
    <width>90</width>

```

```
<height>36</height>
</image>
```

```
<item>
<title>Title Item</title>
<link>http://webpage.uk</link>
<description>
Description
By Author, Date
</description>
</item>
<item>
<title>Title Item 2</title>
<link>http://webpage.uk</link>
<description>
Description 2
By Author, Date
</description>
</item>
</channel>
</rss>
```

3.2 Software Requirements Specifications

3.2.1 Introduction

3.2.1.1 Purpose

The purpose of this document is to describe the features of the Virtual Trade. Users can register in the system and they can buy and sell stocks. The virtual system will permit practise and learn how work the trade without real money.

3.2.1.2 Project features

The product is restricted to beginners in the trade. Users can see the variations of the prices of stocks in the trade by means of graphics. Also, users can see their own stocks and they can buy and sell stocks.

Users will be able to see all information anywhere because the application is in a server connected in the Internet. Henceforth, users can consult the information and do changes when and where they want.

The application is programmed using HTML, PHP, JavaScript and XML to do the RSS.

Moreover, users can export a PDF file with their stocks and their personal information. Furthermore, users can receive RSS about the new purchases and sales.

The system works in any Operative System and browser.

3.2.1.3 Definitions

PDF: (Portable Document Format) is a file format created by Adobe Systems for document exchange. PDF is used for representing two-dimensional documents in a manner independent of the application software, hardware, and operating system. The documents in a PDF format have the smaller in size. PDF is an open standard.

RSS: is a family of Web feed formats used to publish frequently updated works in a standardized format. They benefit readers who want to subscribe to timely updates from favoured websites or to aggregate feeds from many sites into one place. RSS feeds can be read using software called an "RSS reader", "feed reader", or "aggregator", which can be web-based, desktop-based, mobile device or any computerized Internet-connected device. A standardized XML file format allows the information to be published once and viewed by many different programs.

PHP: is a scripting language originally designed for producing dynamic web pages. It has evolved to include a command line interface capability and can be used in standalone software and graphical applications.

PHP was created by Rasmus Lerdorf in 1995. PHP is produced by **The PHP Group**. PHP is free software released under the PHP License. However it is incompatible with the GNU General Public License (GPL), due to restrictions on the usage of the term *PHP*.

PHP is a widely-used general-purpose scripting language that is especially suited for web development and can be embedded into HTML. It generally runs on a web server, taking PHP code as its input and creating web pages as output. It can be deployed on most web servers and on almost every operating system and platform.

HTML (HyperText Markup Language): is the predominant mark-up language for web pages. It is used to describe the structure and the content in a text manner and to complete the text with objects like images. HTML is written in the form of tags that are surrounded by angle brackets (<,>). HTML can also describe the appearance of a document, and can include a script (such as JavaScript) that can affect the behaviour of Web browsers and other HTML processors.

HTML also is used to refer to the MIME type content text/html or XML (like XHTML 1.0 or subsequent versions) or SGML (like HTML 4.01 and previous versions).

HTML files use the extension .htm or .html.

JavaScript: is an interpret language program, it does not require compilation. It is usually used in web pages. It has syntax similar to Java and C language.

JavaScript is an object oriented language like Java. It includes inheritance using the Prototype-based programming, the new classes are generated cloning the base classes (prototypes) and extending their functionality.

All of the modern browsers interpret JavaScript code integrated in the web pages. It works with a DOM implementation.

JavaScript was invented by Brendan Eich in the company Netscape Communications. This company developed the first commercial web browsers. The first time that appeared JavaScript was in the Netscape product Netscape Navigator 2.0.

Years before, it was using in a HTML web pages, to realize tasks and operations in the framework of a customer application, without access to server functions.

JavaScript is executed in the user agent and the sentences are downloading with the HTML code.

Initially, the authors called JavaScript as Mocha and later LiveScript but it was called finally JavaScript in an advertisement of Sun Microsystems and Netscape in 4th December of 1995.

In 1997 the authors proposed JavaScript like a standard. It was an ECMA standard with the EcmaScript name and later it was also ISO standard.

Jscript is the implementation of ECMAScript of Microsoft, very similar to JavaScript of Netscape, but with differences in the object model of the browser that this that the two versions incompatibles.

To avoid the incompatibility, the World Wide Web Consortium designed the Document Object Model standard (DOM), that Konqueror, the latest versions of Internet Explorer, Netscape Navigator, Opera version 7 and Mozilla include it.

DOM (Document Object Model): is a cross-platform and language convention for representing and interacting with objects in HTML, XHTML and XML documents. The rules for programming and interacting with the DOM are specified in the DOM Application Programming Interface (API).

The programs can access and modify the content, structure and style in the HTML and XML documents with DOM.

W3C (World Wide Web Consortium) is the responsible of DOM.

W3C (World Wide Web Consortium): is an international consortium that produces standards to the World Wide Web. Tim Berners-Lee manages W3C, he was also the creator of URL (Uniform Resource Locator), HTTP (HyperText Transfer Protocol) and HTML (Hypertext Mark-up Language). These technologies are the principles in the web.

3.2.2 System features

The product is a web-based software system for virtual stock trading.

3.2.2.1 Functionalities

Database: the system keeps the information about users, companies, orders and stock's prices.

User management: users can register in the system and they can edit their personal information: name, address, age.

Company management: the information about companies (name, address, Website, total) is kept in the system.

Stock management: the system keeps each the stocks with his owner.

Order management: the system keeps the orders to buy and sell. Sellers can order sell stocks and buyers can order buy stocks.

Consult options: users can consult their stocks with the history of the prices.

3.2.2.2 Function requirements

Register: users can register on the system by giving their details, name, address and age.

Open session: users can open session to see more options.

Close session: users can close session.

See information about a specific ticker: both *logged users* and *anonymous users* can see the information about a specific ticker to see their evolution of prices in graphics.

See orders: users can see a list of orders with their prices and details.

See tickers: users can see a list of tickers with their prices and details.

Sell stocks: sellers can sell their stocks. They have to detail the price and the volume.

Buy stocks: buyers can buy their stocks. They have to detail the price and the volume.

3.3 Design

The design of this project is based in the UML diagram Case Uses and entity relationship model (ERM). Moreover, I will explain the structure of the directories in the application, the interface and I will describe how the project works with examples.

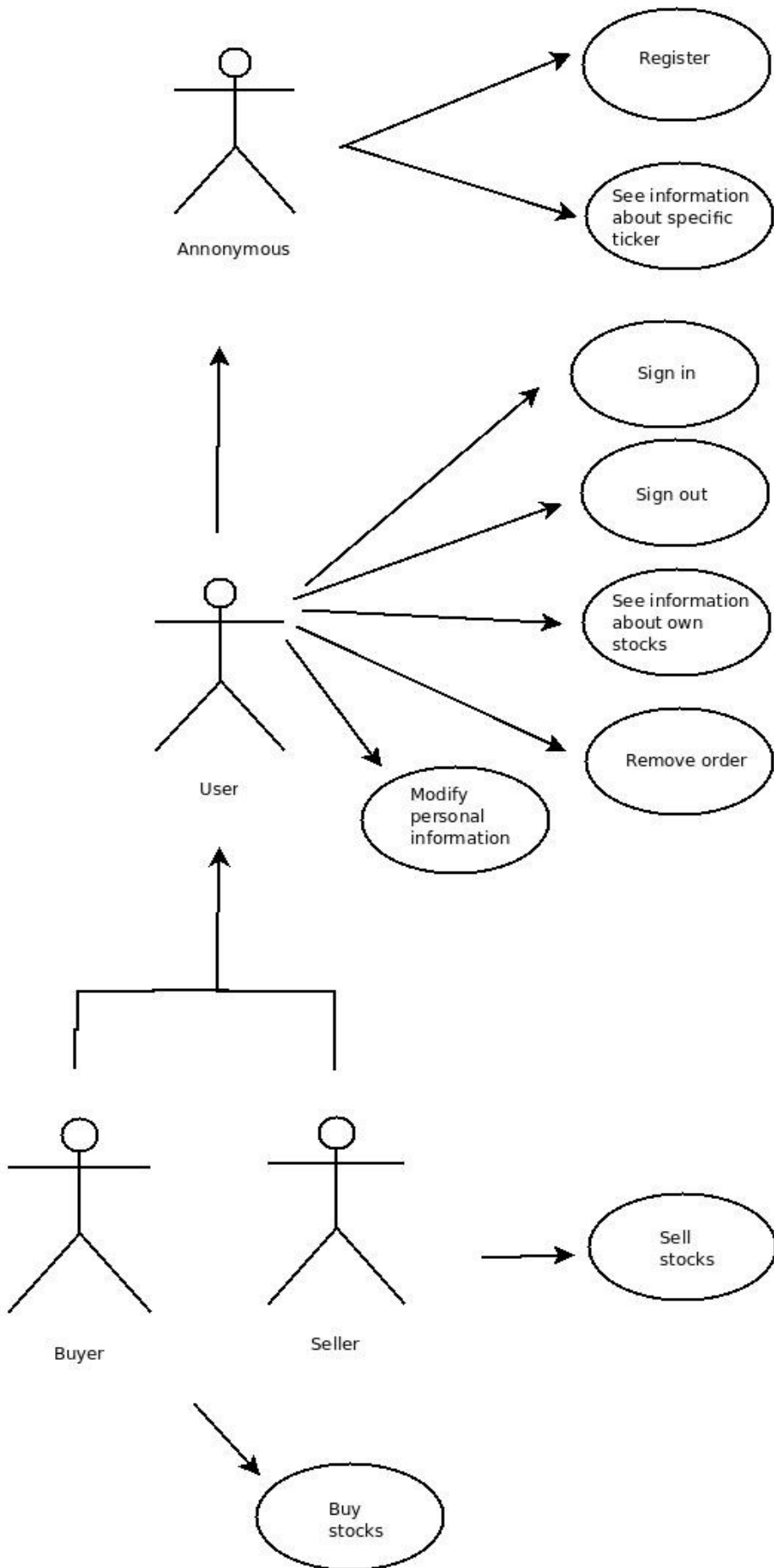
3.3.1 Use Case Diagram

The use case diagram is one of the diagrams of in the Unified Modeling Language (UML) is a type of behavioural diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals and any dependencies between those use cases.

The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be represented.

In this project the case uses diagram is based in the several function of the application like register, sign in and sign out, see details, modify data or sell and buy stocks.

You can see the diagram in the next page of this document.



As you can see in the diagram before, the actor is the user that inherits the anonymous user's use cases. Moreover, you can see the buyer user and seller user that inherit the user's use cases.

Anonymous user has the use cases "register" and "see information about specific ticker" that I am going to describe as follows:

Register

Use Case Description	
Summary	User is registered on the system.
Actors	Anonymous user

Scenario
1) User goes to the register option. System loads the option.
2) User writes login, password, name, address and age that he wants and clicks register. System checks if the introduced login already exists. If the login is repeated the user has to introduce another login. Else the user is registered.

See information about specific ticker

Use Case Description	
Summary	User sees information about the ticker that he wants.
Actors	Anonymous user

Scenario
1) User goes to the "see information about specific ticker" option of the stock that he wants to see. System loads the information of the wished stock.
2) User will be able to choose the way to see the information, per days, per months and per years. System will show the information with the option of the user.

Registered user has the case uses: sign in, sign out, see information about own stocks, remove order and modify personal information.

I will describe this case uses:

Sign in

Use Case Description	
Summary	User signs in the application
Actors	User

Scenario
1) User goes to the “sign in” option. System loads the option.
2) User writes his login and password. System checks if the introduced login already exists and if the pass corresponds with the login. If the login and pass are not ok, the user has to introduce again his login and password. Else the user is logged.

Sign out

Use Case Description	
Summary	User signs out the application
Actors	User

Scenario
1) User goes to the “sign out” option. System loads the options, the information about the user disappears.

See information about own stocks

Use Case Description	
Summary	User sees information about own stocks in the application
Actors	User

Scenario
1) User goes to the “see information about own stocks” option. System loads the information about the current stocks that the user has, the historical transactions that the user did and the personal data of the user.

Remove order

Use Case Description	
Summary	User can remove one order that he did.
Actors	User

Scenario
1) User goes to the “remove order” option of a specific order. System removes the order in the Database.

Modify personal information

Use Case Description	
Summary	User modifies his personal information
Actors	User

Scenario
1) User goes to the “modify personal information” option. System loads “modify personal information” option.
2) User writes the data that he wants in the corresponding place. System uploads the information in the Data Base.

Buyer user has the buy use case.

Buy

Use Case Description	
Summary	User inserts the data that he wants to buy
Actors	Buyer

Scenario
1) User goes to the “buy” option. System loads “buy” option.
2) User inserts the ticker, price and volume that he wants to buy. System saves the data in the Data Base.

Seller user has the sell use case.

Sell

Use Case Description	
Summary	User inserts the data that he wants to sell
Actors	Buyer

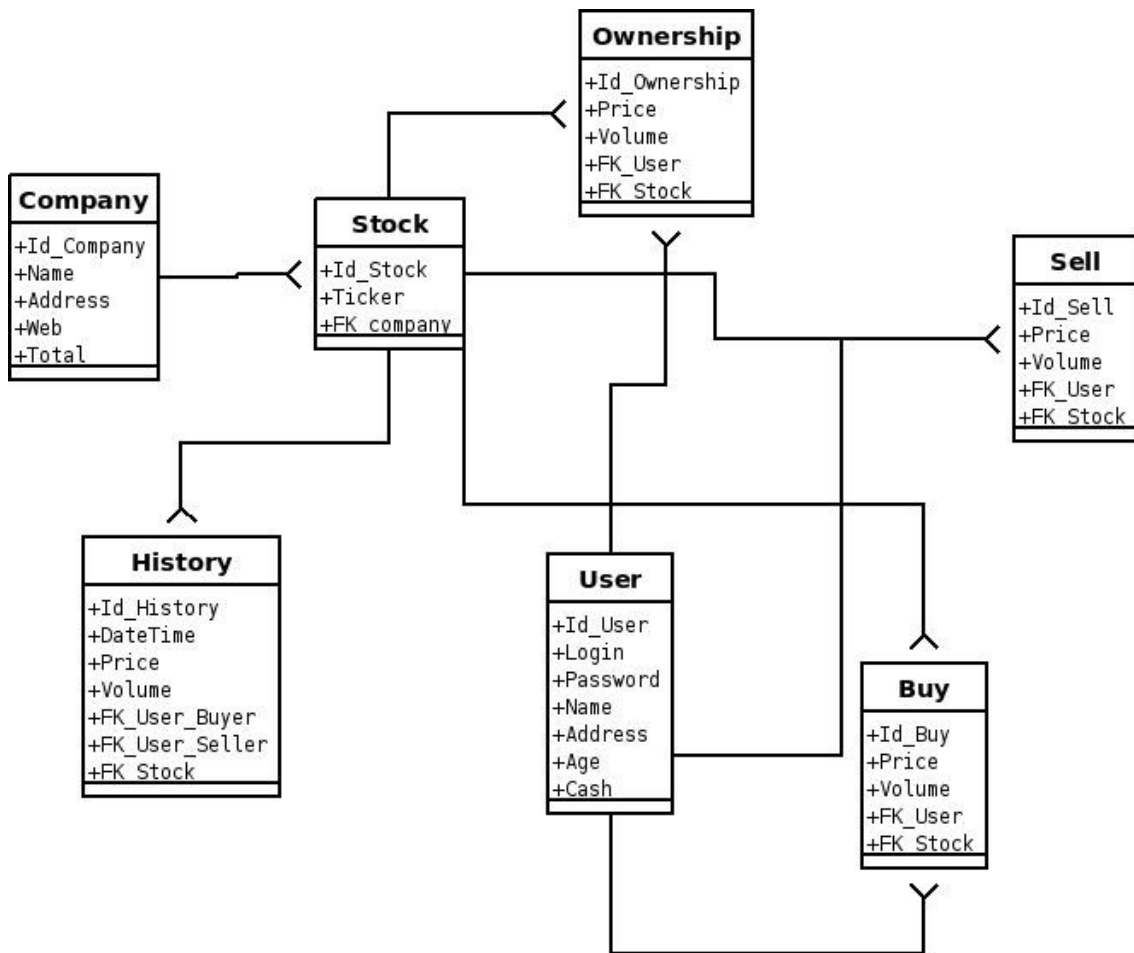
Scenario
1) User goes to the “sell” option. System loads “sell” option and identify the ticker that the user wants to sell.
2) User inserts the price and volume that he wants to sell. System checks if the user has enough stocks of this ticker with the introduced data. If is ok the system saves the data. Else the user has to introduce the price and volume again.

3.3.1 Entity-Relationship Model

An Entity-Relationship Model (ERM) is an abstract and conceptual representation of data. Entity-relationship modelling is a database modelling method, used to produce a type of conceptual schema or semantic data model of a system, often a relational database, and its requirements in a top-down fashion.

In this project I did the Entity-Relationship Model with the relationship of the tables in the Database.

You can see 7 tables, “Ownership”, “Company”, “Stock”, “History”, “User”, “Buy” and “Sell”. “Stock” has a foreign key of “Company”, “Ownership” has a foreign key of “Stock” and another foreign key of “User”, “Sell” has a foreign key of “User” and a foreign key of “Stock”, “Buy” has a foreign key of “User” and a foreign key of “Stock”, “History” has two foreign key of “User” and a foreign key of “Stock”.



In the following paragraph I will explain the tables of the Data Base:

- **Company:** this table contains the information of the companies, name address, web and the total of stocks that the company has.
- **Stock:** this table contains the name of the ticker in each stock.
- **User:** this table contains the information about the users, login, password, name, address, age and the cash that the user has.
- **Ownership:** this table links “User” with “Stock”. It contains the stocks of each user.
- **Sell:** this table links “User” with “Stock”. It contains the stocks that the user wants to sell.
- **Buy:** this table links “User” with “Stock”. It contains the stocks that the user wants to buy.
- **History:** this table links “User” with “Stock”. It contains all the transactions done in the trade market, it has the information about the buyer, seller, number of stocks sold, price, ticker and date.

3.3.2 Directories

This project has several directories to organize the files. You need to execute the file called UniController.exe to run the server. In the */www* and */functions* directory you can find the PHP pages of the virtual trade.

In */www* directory:

- **index.php:** is the main page.
- **buyuser.php:** has the form to buy.
- **details.php:** shows the information about an specific ticker.
- **error_login.php:** shows wrong user when the user inserts the login or password wrong,
- **infouser.php:** shows the information of the user, stocks, buy and sell orders, personal data and transaction history.
- **infouser2.php:** executes the queries to update the personal information of the user.
- **loginindex.php:** has the form to log in.
- **register.php:** has the form to register.
- **register2.php:** executes the queries to register the user in the system.

Inside */www* directory you can find the */functions* directory with the following files:

- **buy.php:** is executed when user presses the link “buy”. It does the queries to buy.
- **buyinyourchoice.php:** is executed when user submits the form to buy in your choice.
- **sell.php:** is called when user presses the link “sell”.
- **sell2.php:** executes the queries to sell.
- **connexion.php:** connects with the data base.

- **divs.php**: has the functions of *printuser()*, *printsellbuyuser()*, *printlogin()*, *printsells()*, *printbuys()*, *printstocks()*.
- **generatepdftables.php**: is called when the user wants to generate a PDF with his information.
- **login.php**: is called when the user submits the form to log in.
- **logout.php**: is called when the user presses the link “logout”.
- **removebuy.php**: is called when the user presses the link “remove” in the buy orders of the user.
- **removesells.php**: is called when the user presses the link “remove” in the sell orders of the user.
- **rss.php**: is called when user presses the RSS option.

```

/virtualtrade
  /DB.txt
  /UniServer
    /disk_start.vbs
    /Server_Start.bat
    /Stop.bat
    /UniController.exe
    /udrive
      /usb_server_start.bat
      /usb_server_stop.bat
      /cgi-bin
      /docs
      /etc
      /home
      /htpasswd
      /plugins
      /ssl
      /tmp
      /usr
      /www
        /.htaccess
        /index.php.bak
        /favicon.ico
        /buyuser.php
        /details.php
        /error_login.php
        /index.php
        /infouser.php
        /infouser2.php
        /loginindex.php
        /register.php
        /register2.php
        /functions
          /buy.php
          /buyinyourchoice.php
          /conexion.php
          /divs.php
          /generatepdftables.php
          /login.php
          /logout.php
          /removebuy.php
          /removesells.php
          /rss.php
          /sell.php
          /sell2.php

```

```

/css
    /style.css
/images
    /money.png
    /title.png
/fpdf16
    /FAQ.htm
    /histo.htm
    /install.txt
    /license.txt
    /fpdf.css
    /fpdf.php
    /doc
    /font
    /tutorial
/graphs
    /graphic.php
    /QPL.txt
    /README
    /VERSION
    /docs
    /src

```

3.3.3 Interface

The interface in this project is very simple and easy to use. You can see in the following image in the upper zone the logo with a hand with bills in the left, in the middle the title “Virtual Trade” and in the right two links, one for the register and the other one to sign in. The down zone has the form to login and a table with the information about the trade, you can see the company, ticker, the change per cent respect the price in the last close, the current price, the volume of stocks sold in the day, the price in the previous close and the date of the last sale.



[Register](#) [Sign in](#)

Identification

Login:

Pass:

Company	Ticker	Change%	Price	Volume	Change	Prev Close	Date
Scotland Bank	SCO	-20%	4	1	-1£	5£	2009-05-07 21:41:49
Barclays	BAR	100%	4	2	2£	2£	2009-05-07 21:41:47
Lloyds	LLO	0%	1	1	0£	1£	2009-04-24 12:56:25
Abbey	ABB	0%	5	5	0£	5£	2009-04-22 20:06:49

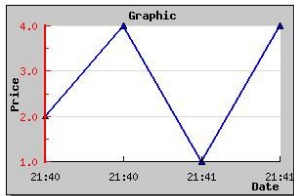
If you click in one ticker you can see the details, for example if you click SCO:



Virtual Trade

SCO

Company	Ticker	Change%	Price	Volume	Change	Prev Close	Date
Scotland Bank	SCO	-20%	4	1	-1£	5£	2009-05-07 21:41:49



Today

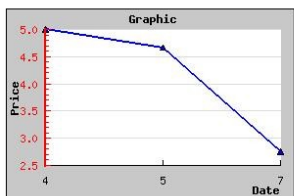
In this page you can change the visual options of the graphic, in stead of view the graphic today you can change it of this month or this year.



Virtual Trade

SCO

Company	Ticker	Change%	Price	Volume	Change	Prev Close	Date
Scotland Bank	SCO	-20%	4	1	-1£	5£	2009-05-07 21:41:49



This month

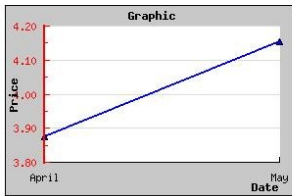
Here, it shows the graphics of this month.



Virtual Trade

SCO

Company	Ticker	Change%	Price	Volume	Change	Prev Close	Date
Scotland Bank	SCO	-20%	4	1	-1£	5£	2009-05-07 21:41:49



This year ▾

And here this year.

If you click the link “Register” you can see the following page:



Virtual Trade

Login:
Password:
Name:
Address:
Age:

[Back to index](#)

You can register in the system, you need to introduce the login, password, name, address, age and click in the bottom “Register”. The system checks if all the fields are full, if it is not the case the system shows a message to the user. The system also checks if the login already exists, if this is the case, the system show a message to the user that the login exists and he has to introduce another one, if is not the case the system registers the new user and saves the data.

If you click on “Sign” in you can see this:



Virtual Trade

Identification

Login:

Pass:

[Back to index](#)

The user has to introduce his login and his password and clicks in the bottom “Access”. The system checks if the login and password are corrects, if it is the case the user can see his information and he can see more options in the system, if it is not the case the system shows a message to the user that he has to put again his login and his password.

If you log in the application you can see more options:



merc

Cash: 866

Stocks			
Ticker	Volume	Price	
SCO	2	4	Sell
SCO	5	1	Sell
SCO	3	5	Sell
BAR	2	4	Sell
BAR	2	5	Sell

Orders			
Sells			
Stock	Price	Volume	
SCO	6	1	Remove
Buys			
Stock	Price	Volume	
BAR	4	6	Remove

[Generate PDF with your information](#)

[Logout](#)

Company	Ticker	Change%	Price	Volume	Change	Prev Close	Date
Scotland Bank	SCO	-20%	4	1	-1£	5£	2009-05-07 21:41:49
Barclays	BAR	100%	4	2	2£	2£	2009-05-07 21:41:47
Lloyds	LLO	0%	1	1	0£	1£	2009-04-24 12:56:25
Abbey	ABB	0%	5	5	0£	5£	2009-04-22 20:06:49

Sells

Stock	Price	Volume	User	
BAR	5	1	Stanley	Buy

[Buy in your choice](#)

Buys

Stock	Price	Volume	User
LLO	2	2	Stanley

In this page you can see in the left part your actual cash, your stocks with the ticker, number of stocks and price that was bought. You can see also your orders in the left part and the orders of the other users in other tables in the middle of the page.

The user can remove his orders clicking in the link near the stocks that he wants to remove, the user can sell also his stocks pressing the link “Sell”, and buy stocks pressing “Buy” in a specific ticker or “Buy in your choice” if the user wants to buy another thing that it is not in the table “Sells”.

User can also log out pressing the link “log out”.

Moreover, user can generate a PDF file clicking in the link “Generate PDF with your information”, and for example for this user the PDF file will be this:

Information of the user
Login: merc
Name: Mercedes Garcia Martinez
Address: Abbey Road 8
Cash: 866

Ticker	Volume	Price
SCO	2	4
SCO	5	1
SCO	3	5
BAR	2	4
BAR	2	5

The PDF file is generated with the personal information of the user and his stocks with their details.

The user, when wants to sell has to field the following things:



[merc](#) [Sign out](#)



You are going to sell:

Ticker:

Price of each stock:

Volume of stocks: (you can sell no more than 9)

The system asks user the price and the volume of stocks that wants to sell. Furthermore, the system informs user how many stocks he can sell, if the user introduce more stocks that he can sell the system sends a message to say that he can not sell this number of stocks and the user has to put other number. In other case the information is saved in the table "Sell".

The user when wants to buy in his choice sees this page:



Virtual Trade

Buy

Buy

Ticker:

Volume:

Price:

The user has to choose the ticker that wants to buy and he has to introduce how many stocks he wants to buy and which price.

Furthermore, if user clicks in the link with his login in the upper part of the page, he can see his information:

merc

Password:

Name:

Address:

Age:

merc

Cash: 866

Stocks			
Ticker	Volume	Price	
SCQ	2	4	Sell
SCQ	5	1	Sell
SCQ	3	5	Sell
BAR	2	4	Sell
BAR	2	5	Sell

Orders			
Sells			
Stock	Price	Volume	
SCQ	6	1	Remove
Buys			

Transaction history

Purchases

Ticker	Price	Volume	Time
LLO	5	0	2009-04-18 17:13:38
LLO	5	5	2009-04-19 18:11:19
BAR	5	2	2009-03-22 12:55:27
BAR	5	5	2009-04-22 14:21:14
BAR	2	3	2009-02-22 17:26:20
BAR	2	2	2009-02-22 17:26:22
SCQ	4	5	2009-04-22 18:08:40
LLO	1	5	2009-04-22 18:08:42
ABB	2	5	2009-04-22 18:08:44
ABB	3	5	2009-04-22 19:14:54
ABB	5	5	2009-04-22 20:06:49
LLO	3	3	2009-04-23 16:28:31
LLO	2	2	2009-04-23 17:33:52
BAR	5	5	2009-04-23 18:42:59
SCQ	5	1	2009-05-04 23:05:53
SCQ	5	1	2009-05-04 23:21:29
SCQ	5	2	2009-05-04 23:32:12
SCQ	5	2	2009-05-05 09:27:06
SCQ	5	1	2009-05-05 10:12:00

Sales

Ticker	Price	Volume	Time
LLO	5	0	2009-04-18 17:45:41
SCQ	2	0	2009-04-18 16:53:30
SCQ	5	0	2009-04-19 16:54:53
LLO	5	0	2009-04-18 17:12:58
SCQ	6	0	2009-04-19 17:33:18
BAR	5	5	2008-04-17 18:11:51
BAR	6	2	2009-03-19 18:17:22
BAR	7	1	2009-04-22 17:07:54
LLO	5	5	2009-04-22 17:07:57
BAR	2	1	2009-04-20 17:07:59
ABB	5	5	2009-04-22 19:10:49
ABB	5	5	2009-04-22 19:45:32
LLO	6	5	2009-04-22 19:46:29
SCQ	3	5	2009-04-22 20:01:26
LLO	3	5	2009-04-22 20:10:26
LLO	4	5	2009-04-23 16:28:59
SCQ	5	2	2009-04-23 17:25:22
LLO	1	1	2009-04-24 12:53:43
LLO	1	1	2009-04-24 12:56:25

The user can see his personal data and can modify this. Moreover, he can see his stocks, orders and transaction history (all the sales and purchases that he did in all the history of his login).

3.3.4 Example of a Demonstration

I will show an example of transaction. If we have the system like the following image:



Stanley

Cash: 924

Stocks			
Ticker	Volume	Price	
SCO	1	6	Sell
BAR	1	5	Sell
BAR	1	3	Sell
BAR	1	4	Sell

Orders			
Sells			
Stock	Price	Volume	
Buys			
Stock	Price	Volume	
LLO	2	2	Remove

[Generate PDF with your information](#)

[Logout](#)

Company	Ticker	Change%	Price	Volume	Change	Prev Close	Date
Scotland Bank	SCO	50%	6	1	2£	4£	2009-05-08 12:51:51
Barclays	BAR	25%	5	1	1£	4£	2009-05-08 12:51:27
Lloyds	LLO	0%	1	1	0£	1£	2009-04-24 12:56:25
Abbey	ABB	0%	5	5	0£	5£	2009-04-22 20:06:49

Sells

Stock	Price	Volume	User

[Buy in your choice](#)

Buys

Stock	Price	Volume	User
BAR	4	6	merc

Stanley signed in, he has an order that he wants to buy 2 stocks of LLO for £2. Now, Stanley decides to sell 1 stock of SCO for £5. The system is now of this way:



Stanley

Cash: 924

Stocks			
Ticker	Volume	Price	
SCO	1	6	Sell
BAR	1	5	Sell
BAR	1	3	Sell
BAR	1	4	Sell

Orders			
Sells			
Stock	Price	Volume	
SCO	5	1	Remove
Buys			
Stock	Price	Volume	
LLO	2	2	Remove

[Generate PDF with your information](#)

[Logout](#)

Company	Ticker	Change%	Price	Volume	Change	Prev Close	Date
Scotland Bank	SCO	50%	6	1	2£	4£	2009-05-08 12:51:51
Barclays	BAR	25%	5	1	1£	4£	2009-05-08 12:51:27
Lloyds	LLO	0%	1	1	0£	1£	2009-04-24 12:56:25
Abbey	ABB	0%	5	5	0£	5£	2009-04-22 20:06:49

Sells


Stock	Price	Volume	User

[Buy in your choice](#)

Buys

Stock	Price	Volume	User
BAR	4	6	merc

Now, *merc* can see the following image:



merc

Cash: 867

Stocks			
Ticker	Volume	Price	
SCO	1	4	Sell
SCO	4	1	Sell
SCO	2	5	Sell
BAR	3	4	Sell
BAR	3	5	Sell

Orders			
Sells			
Stock	Price	Volume	
Buys			
Stock	Price	Volume	
BAR	4	6	Remove

[Generate PDF with your information](#)

[Logout](#)

Company	Ticker	Change%	Price	Volume	Change	Prev Close	Date
Scotland Bank	SCO	50%	6	1	2£	4£	2009-05-08 12:51:51
Barclays	BAR	25%	5	1	1£	4£	2009-05-08 12:51:27
Lloyds	LLO	0%	1	1	0£	1£	2009-04-24 12:56:25
Abbey	ABE	0%	5	5	0£	5£	2009-04-22 20:06:49

Sells


Stock	Price	Volume	User
SCO	5	1	Stanley

[Buy in your choice](#)

Buys

Stock	Price	Volume	User
LLO	2	2	Stanley

merc decided to buy the order in “Sells”. So, the row in the table “Sells” will disappear, *merc* will have the stock and Stanley will not have the stock. Moreover, the row of SCO in the table of the companies will change. As you can see in the following image, instead of 50% you can see 25%, the price changed to £5, the Change now is £1 and the Date changed too.



merc

Cash: 862

Stocks			
Ticker	Volume	Price	
SCO	2	4	Sell
SCO	5	1	Sell
SCO	3	5	Sell
BAR	3	4	Sell
BAR	3	5	Sell

Orders			
Sells			
Stock	Price	Volume	
Buys			
Stock	Price	Volume	
BAR	4	6	Remove

[Generate PDF with your information](#)

[Logout](#)

Company	Ticker	Change%	Price	Volume	Change	Prev Close	Date
Scotland Bank	SCO	25%	5	1	1£	4£	2009-05-08 13:22:46
Barclays	BAR	25%	5	1	1£	4£	2009-05-08 12:51:27
Lloyds	LLO	0%	1	1	0£	1£	2009-04-24 12:56:25
Abbey	ABE	0%	5	5	0£	5£	2009-04-22 20:06:49

Sells

Stock	Price	Volume	User

[Buy in your choice](#)

Buys

Stock	Price	Volume	User
LLO	2	2	Stanley

Now, *Stanley* decides to buy one ticker BAR for £5:



Virtual Trade

Buy

Buy

Ticker:

Volume:

Price:

They system now looks this:



Stanley

Cash: 929

Stocks			
Ticker	Volume	Price	
BAR	1	5	Sell
BAR	1	3	Sell
BAR	1	4	Sell

Orders			
Sells			
Stock	Price	Volume	
Buys			
Stock	Price	Volume	
BAR	5	1	Remove
LLO	2	2	Remove

[Generate PDF with your information](#)

[Logout](#)

Company	Ticker	Change%	Price	Volume	Change	Prev Close	Date
Scotland Bank	SCO	25%	5	1	1£	4£	2009-05-08 13:22:46
Barclays	BAR	25%	5	1	1£	4£	2009-05-08 12:51:27
Lloyds	LLO	0%	1	1	0£	1£	2009-04-24 12:56:25
Abbey	ABB	0%	5	5	0£	5£	2009-04-22 20:06:49

Sells

Stock	Price	Volume	User
Buy in your choice			

Buy

Stock	Price	Volume	User
BAR	4	6	merc

merc decides sell 2 BAR ticker for £5:



Virtual Trade

You are going to sell:

Ticker:

Price of each stock:

Volume of stocks: (you can sell no more than 6)

The data now is:



merc

Cash: 867

Ticker	Volume	Price	
SCO	2	4	Sell
SCO	5	1	Sell
SCO	3	5	Sell
BAR	2	4	Sell
BAR	2	5	Sell

Company	Ticker	Change%	Price	Volume	Change	Prev Close	Date
Barclays	BAR	25%	5	1	1£	4£	2009-05-08 13:44:41
Scotland Bank	SCO	25%	5	1	1£	4£	2009-05-08 13:22:46
Lloyds	LLO	0%	1	1	0£	1£	2009-04-24 12:56:25
Abbey	ABB	0%	5	5	0£	5£	2009-04-22 20:06:49

Orders

Sells

Stock	Price	Volume	User
BAR	5	1	Remove

Buy in your choice

Buys



Stock	Price	Volume	User
BAR	4	6	Remove

Generate PDF with your information

Logout

merc sold 1 BAR ticker for £5 to Stanley, now she wants to sell one more for £5. The row of Barclays in the table of “Companies” only changed the Date because the other data was the same than the last sale.

You also can see the details today of the BAR ticker:

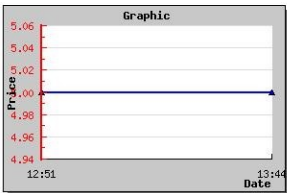



[merc](#) [Sign out](#)

BAR

Company	Ticker	Change%	Price	Volume	Change	Prev Close	Date
Barclays	BAR	25%	5	1	1£	4£	2009-05-08 13:44:41

Graphic



Price: 5.06, 5.04, 5.02, 5.00, 4.98, 4.96, 4.94

Date: 12:51, 13:44

Today

The line is flat because the two sales today were sold for the same price.

3.4 Implementation details

3.4.1 Tools

I used different tools to do the project, Macromedia Dreamweaver to program the code, Uniform server, Apache, MySQL and PHP.

Macromedia Dreamweaver MX

I needed a website editor to do the pages of the program. I have used Adobe Dreamweaver, created by Adobe (previously Macromedia). It is the website editor more used in the web and programming's sector for his functionalities. It has support for image editing to animation through its integration with other tools.

Dreamweaver creates and administrates professional websites and powerful Internet applications. I was achieved for the first time in order to facilitate a working environment that can create, make and manage Web sites and applications quickly using visual composition tools, rapid application development Internet and extensive support for code editing.

It can also be used as a professional code editor for HTML visual design and administration of Web sites and pages. It allows us both control HTML code manually, such as working in a visual editing environment, and providing tools for better Web design. In addition, technology Macromedia Roundtrip HTML imports HTML documents without the need for reformat the code and you can set Dreamweaver to clean and change the HTML formats as desired.

Until the MX version, Dreamweaver was severely criticized for its lack of support Web standards, because the code it generated was often only valid for Internet Explorer, and does not validate as standard HTML. This has been corrected in recent versions.

The main advantage of this editor is that its routines (such as inserting a hyperlink or picture) are in JavaScript-C. This means the program files are not instructions but C + +, JavaScript routines that makes them very fluid.

The original versions of the application were used as mere editors WYSIWYG (editors that show what we're doing at the time), but newer versions support other web technologies CSS, JavaScript and some server-side frameworks.

Dreamweaver has been a success since the late 90's and now keeps 90% of the HTML editors. This application is available for both MAC and Windows platform, but can also run on UNIX-based platforms using emulators such as Wine.

It also permits the user to use the most of the Web browsers installed on their computer to preview websites. It also has site management tools aimed at beginners, for example, the ability to find and replace lines of text and code specified for any parameter to the entire website. The panel behaviour also allows you to create basic JavaScript without any knowledge of code.

With the advent of version MX Macromedia incorporated tools creating dynamic content in Dreamweaver. In essence the HTML WYSIWYG tools, it also allows connecting to databases such as MySQL and Microsoft Access to filter and display content using technology script such as ASP (Active Server Pages), ASP.NET, ColdFusion, JSP (JavaServer Pages), PHP without having prior experience in programming.

Dreamweaver MX offers improved support for Cascading Style Sheets. Dreamweaver can display most CSS properties, while a revised CSS Panel simplifies the process of applying and editing styles to text, images, and links. *Dreamweaver MX [3]*.

One aspect of high Dreamweaver is its extensible architecture. It allows the use of "extensions". The extensions, as there are known to us, are small programs, which any web developer can write (usually in HTML and JavaScript), download and install, thereby providing added functionality to the application. Dreamweaver is support of a large community of developers who make extensions the availability of free and paid extensions for most of the web development tasks, ranging from simple rollover effects to full Shopping cart *Macromedia Dreamweaver MX 2004*

Mozilla Firefox

I used a browser to test the project, you need a browser to use the application. I chose Mozilla Firefox but you can use whichever browser that you want to.

Firefox is a browser developed by the Mozilla Corporation and a large number of externs volunteers. It started as a derivative of the Mozilla Application Suite, which eventually replace it as the best product of the Mozilla project. Firefox is available for Microsoft Windows, Mac OS X and GNU / Linux. In addition, the code has been modified by others to work on other UNIX operating systems.

The Firefox source code is freely available under the tri-license Mozilla as a free and open source. Being free the community is always ready to enhance their development. You can run only under Windows but not on nearly all platforms. It is widely used in the Linux world where a majority. The current stable version is 3.0.10.

Firefox has become one of the most downloaded free applications, especially among home users with 25 million downloads. The first 99 days following the release of the first version on 19 October 2005, Firefox had reached 100 million downloads less than one year (344 days). Version 1.5 came November 29 2005 exceeded the height of the 2 million downloads in the first 36 hours. By August 2006 the stats had exceeded 200 million [Montenegro, 2006].

The novel features incorporated in Firefox were the following:

- Pop-up blocking.
- The tabbed browsing.
- Dynamic bookmarks.
- Support for open standards.
- Custom themes.
- Internal search engine to find a word in the current page.

- Support of flow RSS. RSS is part of the family of formats XML have been developed specifically for weblogs and sites news that are updated frequently. So you can share information and use it on other sites or programs.
- Options for proper administration in the preferences in regard to confidential data such as passwords or the fields to fill in forms.

Firefox supports the W3C standards (World Wide Web Consortium). This consortium is responsible for defining the rules of the Web, by issuing various recommendations that enable developers to create Web sites as optimized potential *Firefox-Thunderbird [10]*.

Uniform server

I installed a server called Uniserver version 4.0 which has incorporated PHP, MySQL and Apache.

The Uniform Server is a WAMP package that allows you to run a server on any Microsoft System Windows Operative System based computer. It is easy to download or move around and can also be used or setup as a production/live server. Developers also use The Uniform Server to test their applications made with either PHP, MySQL, Perl, or the Apache HTTPd Server.

There are some featured products that use The Uniform Server as a backbone, or as a part of the software. You can read the list below:

- JSAS (Joomla Stand Alone Server)
- MSAS (Mambo Stand Alone Server)
- XSAS (Xoops Stand Alone Server)

I took this information from the website of Uniform Server: <http://www.uniformserver.com/> [15].

Apache

The server used in this project is Apache.

The whole purpose of a web server is to translate a URL either into a filename, and then send that file back over the Internet, or into a program name, and then run that program and send its output back.

Apache has more than twice the market share than its other competitor, Microsoft. This is not just because it is freeware and costs nothing. It is also an open source, which means that the source code can be examined by anyone.

Apache is a program that runs under a suitable multitasking operating system. In this project we use Apache with Windows.

Apache gets its name from the fact that it consists of some existing code plus some patches.

Apache is free and was written by a team of volunteers who do not get paid for their work.

The first web server was built by the British physicist Tim Berners-Lee at CERN, the European Centre for Nuclear Research at Geneva, Switzerland. The immediate ancestor of Apache was built by the U.S. government's NCSA, the National Center for Supercomputing Applications. *Apache. The definitive Guide [4]*

MySQL

MySQL is a free database management system like mSQL, Postgres. The most attractive features of MySQL's features are:

- Speed.
- Ease of use. MySQL is less complex than other larger systems.
- Cost. MySQL is free.
- Query language support. MySQL understands SQL, the most widely used language for the modern database systems. You can also access MySQL using applications that support ODBC, developed by Microsoft.
- Capability. Many clients can connect to the server at the same time. You can access MySQL interactively using several interfaces that let you enter queries and view the results: command-line clients, Web browsers, or X Windows System clients. In addition, a variety of programming interfaces is available for languages such as C, Perl, Java, PHP, and Python.
- Connectivity and security. MySQL is networked, and databases can be accessed from anywhere on the Internet, so you can share your data with anyone. But MySQL has access control for security.
- Portability. MySQL runs on many varieties of UNIX, and other non-UNIX systems like Windows and OS/2. MySQL runs on hardware from home PCs to high-end servers.
- Open distribution. To obtain MySQL you only have to download it from the Internet and you can see and change the code. *[6] MySQL.*

MySQL uses SQL language. The structured query language SQL (Structured Query Language) is a declarative language to access relational databases that allows you to specify various types of transactions at the same. One of its features is the handling of relational algebra and the calculation can launch queries to retrieve, in a simple-interest information from a database, and also make changes on it. It is a fourth generation language (4GL).

SQL is a language access to databases that exploits the flexibility and power of relational systems enabling a variety of operations on them.

Declarative language is a "high level" or "not applicable", which thanks to its strong theoretical foundation and guidance to the handling of sets of records, not individual records, allows high productivity in coding and object-orientation. In this way a single statement may amount to one or more programs you can use on a low-level language-oriented record. *<http://wikipedia.org> [24].*

PHP

PHP is a server-side scripting language designed specifically for the Web. Within an HTML page, you can embed PHP code that will be executed each time the page is visited. Your PHP code is interpreted at the web server and generates HTML or other output that the visitors will see.

PHP was conceived in 1994 and was originally the work of Rasmus Lerdorf.

PHP is an Open Source product, which means you have access to the source code and can use, alter, and redistribute it all without charge.

PHP originally stood for Personal Home Page but was changed in line with the GNU (GNU = Gnu's Not Unix) recursive naming convention and now stands for PHP Hypertext Preprocessor.

Some of PHP's features are listed below:

- **High performance:** PHP is very efficient. You can serve millions of hits per day.
- **Database Integration:** PHP has native connections available to many database systems. In addition to MySQL, you can directly connect to PostgreSQL, mSQL, Oracle, dbm, FilePro, Hyperwave, Informix, InterBase and Sybase databases. PHP 5 also has a built-in SQL interface to a flat file, called SQLite. Using the Open Database Connectivity Standard (ODBC), you can connect to any database that provides an ODBC driver. This includes Microsoft products and many others.
- **Built-in Libraries:** Because PHP was designed for use on the Web, it has many built-in functions for performing many useful web-related tasks. You can generate GIF images on the fly, connect to web services and other network services, parse XML, send email, work with cookies, and generate PDF documents, all with just a few lines of code.
- **Cost:** PHP is free. You can download the latest version from the website.
- **Ease of Learning PHP:** The syntax of PHP is based on other programming languages, primarily C and Perl.
- **Object-Oriented Support:** PHP 5 has well-design object-oriented features. If you learned to program in Java or C++, with PHP you can use inheritance, private and protected attributes and methods, abstract classes and methods, interfaces, constructors, destructors and built-in iteration behaviour.
- **Portability:** PHP is available for many different operating systems. You can write PHP code on free Unix-like operating systems such as Linux and FreeBSD, commercial Unix versions such as Solaris and IRIX, or on different versions of Microsoft Windows.
- **Source code:** You can modify or add something to the language.
- **Availability of Support:** Zend Technologies, the company behind the engine that powers PHP, funds its PHP development by offering support and related software on a commercial basis. *PHP and MySQL* [7].

3.4.2 Code explanation

In this section, important functions like *printlogin()*, *printuser()*, *printsells()*, *printbuys()* and *printstocks()* are explained below :

printlogin()

printlogin() function is called in **index.php**, it prints the form to sign in the program. The form asks the user to enter the login and password details and calls a JavaScript function. The JavaScript function checks if the user has entered the login and password details. Finally, the form calls **login.php** that checks if the login and the password details are corrects

The code is given below:

```
<?
function printlogin()
{
?>

<script type="text/javascript">
function check()
{
  if(document.forms["formLogin"].login.value == "")
  {
    alert("Write your login");
    document.forms["formLogin"].login.focus();
    return false;
  }
  else if(document.forms["formLogin"].password.value == "")
  {
    alert("Write your password");
    document.forms["formLogin"].password.focus();
    return false;
  }
  else
  {
    return true;
  }
}
</script>

<form name="formLogin" method="post" action="functions/login.php">
<table id="identification">
<tr>
<td colspan="2" align="center"><h4>Identification</h4></td>
</tr>
<tr>
<td align="right">Login: </td><td><input type="text" name="login"
/></td>
</tr>
<tr>
<td align="right">Password: </td><td><input type="password"
name="password" /></td>
</tr>
<tr>
```

```

<td colspan="2" align="center"><input type="submit" value="Access"
onClick="return check() "/>
</td>
</tr>
</table>
</form>

<?
}

```

printuser()

printuser() function prints the login and the cash details of the logged user. *printuser()* function also prints the user's stocks and if the user wants to sell some stock he has to just press the link "sell" near the stock. Moreover, shows the intended sales and purchases of the logged in user. The user can remove the selling and buying orders that he does not want, and can press any ticker to see the details of this ticker.

The code is shown below:

```

function printuser(){
?>
<h2>

<?php
echo $_SESSION['login'];
?>

</h2>
Cash:
<?php
$query = "SELECT * FROM User where login='".$_SESSION['login']."'";
$answer = mysql_query($query);
$row = mysql_fetch_array($answer);
echo $row["cash"];
$id = $row["id"];
echo "<br>";

//print the user's stocks
$query = "SELECT Ownership.id AS ownerid, ticker, volume, stock, price
FROM Ownership, Stock where user=".$id." and Ownership.stock =
Stock.id";
$answer = mysql_query($query);
echo "<table border='1'>";
echo "<tr><td colspan='4'><b>Stocks</b></td></tr>";
echo "<tr><td>Ticker</td><td>Volume</td><td>Price</td></tr>";
while($row = mysql_fetch_array($answer)){
    echo "<tr>";
    echo "<td>";
    echo "<a href='details.php?idticker=".$row["stock"]."&ticker="
$row["ticker"]."'>".$row["ticker"]."</a>";
    echo "</td>";
    echo "<td>";
    echo $row["volume"];
    echo "</td>";
    echo "<td>";
}

```

```

        echo $row["price"];
        echo "</td>";
        echo "<td>";
        echo          "<a          href='/functions/sell.php?ownerid=".$row["ownerid"]."&ticker=".$row["ticker"]."&volume=".$row["volume"]."&stock=".$row["stock"]."'>Sell</a>";
        echo "</td>";
        echo "</tr>";
    }
    echo "</table>";

//print the sell intentions of this user
echo "<table border='1'>";
echo "<tr><td colspan='4'><b>Orders</b></td></tr>";
echo "<tr><td colspan='4'>Sells</td></tr>";
echo "<tr><td>Stock</td><td>Price</td><td>Volume</td></tr>";
$query = "SELECT Sell.id AS idsell, ticker, user, price, volume,
stock, price FROM Sell, Stock where user=".$id." AND Stock.id =
Sell.stock";
$answer = mysql_query($query);
while($row = mysql_fetch_array($answer)) {
    echo "<tr>";
    echo "<td>";
    echo "<a href='details.php?idticker=".$row["stock"]."&ticker=".$row["ticker"]."'>".$row["ticker"]."</a>";
    echo "</td>";
    echo "<td>";
    echo $row["price"];
    echo "</td>";
    echo "<td>";
    echo $row["volume"];
    echo "</td>";
    echo "<td>";
    echo          "<a          href='/functions/removesells.php?idsell=".$row['idsell']."'>Remove</a>";
    echo "</td>";
    echo "</tr>";
}
//print the buy intentions of this user
echo "<tr><td colspan='4'>Buys</td></tr>";
echo "<tr><td>Stock</td><td>Price</td><td>Volume</td></tr>";
$query = "SELECT Buy.id AS idbuy, ticker, user, price, volume, stock
FROM Buy, Stock where user=".$id." AND Stock.id = Buy.stock";
$answer = mysql_query($query);
while($row = mysql_fetch_array($answer)) {
    echo "<tr>";
    echo "<td>";
    echo "<a href='details.php?idticker=".$row["stock"]."&ticker=".$row["ticker"]."'>".$row["ticker"]."</a>";
    echo "</td>";
    echo "<td>";
    echo $row["price"];
    echo "</td>";
    echo "<td>";
    echo $row["volume"];
    echo "</td>";
    echo "<td>";
    echo          "<a          href='/functions/removebuy.php?idbuy=".$row['idbuy']."'>Remove</a>";
    echo "</td>";
    echo "</tr>";
}

```

```

}
echo "</table>";
}

```

printsells() and printbuys()

printsells() and *printbuys()* functions are very similar. *printsells()* shows the “Sell” table in order without the current user’s stocks and the logged in users can buy any row by clicking on the link near the rows. *printbuys()* shows the “buy” table in descendant order without the current user’s stocks too. User can see any stock’s details only by clicking on the stock.

The code is given below:

```

function printsells() {
?>
<h2>Sells</h2>
<?
$currentuser = $_SESSION['login'];
$query = "SELECT *, Sell.id AS idbuy, User.login AS loginuser FROM
Sell, User, Stock WHERE Sell.user = User.id and Stock.id = Sell.stock
ORDER BY Sell.price";
$answer = mysql_query($query);
echo "<table border='1'>";
echo
"<tr><td>Stock</td><td>Price</td><td>Volume</td><td>User</td></tr>";
while($row = mysql_fetch_array($answer)) {
    //don't print the current user's sell intentions
    if($row["loginuser"]!=$currentuser) {
        echo "<tr>";
        echo "<td>";
        //User can see the ticker's details
        echo "        <a                href='details.php?idticker=" .
$row["stock"]."&ticker=".$row["ticker"]."'>".$row["ticker"]."</a>";
        echo "</td>";
        echo "<td>";
        echo $row["price"];
        echo "</td>";
        echo "<td>";
        echo $row["volume"];
        echo "</td>";
        echo "<td>";
        echo $row["login"];
        echo "</td>";
        echo "<td>";
        //User can buy this
        echo "        <a                href='/functions/buy.php?idbuy=" .
$row["idbuy"]."&buyer=".$currentuser."'>Buy</a>";
        echo "</td>";
        echo "</tr>";
    }
}
echo "</table>";
}

```

```

function printbuys() {
?>
<h2>Buys</h2>

```

```

<?
$query = "SELECT * FROM User WHERE cash <= 0";
$answer = mysql_query($query);
while($row = mysql_fetch_array($answer)){
    $query = "DELETE FROM Buy WHERE user=".$row['id'];
    mysql_query($query);
}
//Buys table in descendant order
$query = "SELECT * , User.login AS loginuser FROM Buy, User, Stock
WHERE Buy.user = User.id and Stock.id = Buy.stock ORDER BY Buy.price
DESC";
$answer = mysql_query($query);
echo "<table border='1'>";
echo
"<tr><td>Stock</td><td>Price</td><td>Volume</td><td>User</td></tr>";
while($row = mysql_fetch_array($answer)){
    //don't print the current user's sell intentions
    if($row["loginuser"]!=$_SESSION['login']){
        echo "<tr>";
        echo "<td>";
        //User can see the ticker's details
        echo "<a href='details.php?idticker=" .
$row["stock"]."&ticker=".$row["ticker"]."'">".$row["ticker"]."</a>";
        echo "</td>";
        echo "<td>";
        echo $row["price"];
        echo "</td>";
        echo "<td>";
        echo $row["volume"];
        echo "</td>";
        echo "<td>";
        echo $row["login"];
        echo "</td>";
        echo "</tr>";
    }
}
echo "</table>";
}

```

printstocks()

printstocks() function prints information about the stocks. It makes a query with the “History” table in descendant order and also makes a query with the current date.

printstocks() function shows the current price of each ticker with the volume of stocks that were sold, the changes in pounds and the percent respect to the last price in last day that someone bought, the last price and the date of the last transaction.

It used an array called “Companies” that stores the tickers we use in order to not repeat the same ticker in the table shown.

The code is given below:

```

function pricestocks(){
//companies array to don't repeat any company
$companies[] = array();
//query of get the history prices

```

```

$query = "SELECT History.id AS idhistory, History.stock,
History.price, History.time, History.volume, Stock.ticker,
Company.name FROM History, Stock, Company
WHERE History.stock=Stock.id AND Stock.company=Company.id ORDER BY
History.time DESC";
$answer = mysql_query($query);
//query to obtain the current date
$querytoday = "SELECT DISTINCT year(now()) AS year, month(now()) AS
month, day(now()) AS day FROM History";
$answertoday = mysql_query($querytoday);
$rowtoday = mysql_fetch_array($answertoday);
$year = $rowtoday["year"];
$month = $rowtoday["month"];
$day = $rowtoday["day"];
$datetoday = $year."-".$month."-".$day." 00:00:00";
echo "<table border='1'>";
echo "<tr><td>Company</td><td>Ticker</td><td>Change
%</td><td>Price</td><td>Volume</td><td>Change</td><td>Prev
Close</td><td>Date</td></tr>";
while($row = mysql_fetch_array($answer)){
//select the last price of the last day before
$querypricebefore = "SELECT History.price AS pricebefore FROM
History, Stock WHERE History.stock=Stock.id AND time < '".
$datetoday.'" AND History.stock = ".$row["stock"]." ORDER BY
History.time DESC";
$answerpricebefore = mysql_query($querypricebefore);
while($rowpricebefore = mysql_fetch_array($answerpricebefore)){
//if the company is not in $companies
if (!in_array($row["name"], $companies)){
echo "<tr>";
echo "<td>";
echo $row["name"];
echo "</td>";
echo "<td>";
//user can see the ticker's details and send the
data by GET method
echo "<a href='details.php?idticker=" .
$row["stock"]."&ticker=".$row["ticker"]."&name=" .
$row["name"]."&price=".$row["price"]."&pricebefore=" .
$rowpricebefore["pricebefore"]."&volume=".$row["volume"]."&time=" .
$row["time"].">".$row["ticker"]."</a>";
echo "</td>";
echo "<td>";
//change%, compare the current price with last price
in last day
$number = 100*($row["price"]-
$rowpricebefore["pricebefore"])/$rowpricebefore["pricebefore"];
//only 2 decimals
echo round($number,2)."%";
echo "</td>";
echo "<td>";
echo $row["price"];
echo "</td>";
echo "<td>";
echo $row["volume"];
echo "</td>";
echo "<td>";
//subtraction of the current price and the last price in
last day
echo $row["price"]-
$rowpricebefore["pricebefore"]."£";

```



```

$resultowner = mysql_query($query, $conexion);
$rowowner = mysql_fetch_array($resultowner);
//query to know the stock's number that user wants to sell previously
$query = "SELECT SUM(Volume) AS Volsell FROM Sell WHERE stock = ".
$stock." AND user=".$row["id"];
$resultsell = mysql_query($query, $conexion);
$rowsell = mysql_fetch_array($resultsell);

//number of stocks that user can sell
$total = $rowowner["Volowner"] - $rowsell["Volsell"];

?>
<script type="text/javascript">
function check()
{
    if(document.forms["formSell"].price.value == "")
    {
        alert("Write the price");
        document.forms["formSell"].price.focus();
        return false;
    }
    else if(document.forms["formSell"].volume.value == "")
    {
        alert("You have to write the number of stocks that you want to
sell");
        document.forms["formSell"].volume.focus();
        return false;
    }
    else if(document.forms["formSell"].price.value < 0)
    {
        alert("The price can't be negative");
        document.forms["formSell"].price.focus();
        return false;
    }
    else if(parseInt(document.forms["formSell"].volume.value) > <?
echo $total; ?>)
    {
        alert("You can't sell all of this stocks");
        document.forms["formSell"].volume.focus();
        return false;
    }

    else
    {
        return true;
    }
}
</script>
<form name="formSell" method="post" action="sell2.php?ownerid=<? echo
$ownerid; ?>&ticker=<? echo $ticker; ?>&stock=<? echo $stock; ?>">
<table id="wantsell">
<tr>
<td align="left">Price of each stock: </td><td><input type="text"
name="price" /></td>
</tr>
<tr>
<td align="left">Volume of stocks:</td><td><input type="text"
name="volume" /></td><td align="right">(you can sell no more than <?
echo $total; ?>) </td>
</tr>

```

```

<tr>
</tr>
<tr>
<td colspan="2" align="center"><input type="submit" value="Sell"
onClick="return check()"/>
</tr>
</table>
<input type="hidden" name="ownerid" value="<? echo $ownerid; ?>"/>
<input type="hidden" name="stock" value="<? echo $stock; ?>"/>
</form>

</body>
</html>

```

buy.php

This function is called when the user clicks in the link “buy”, the system knows the ticker that the user wants to buy and the purchase is made.

The function conducts the following actions:

- The system updates the cash of the buyer.
- The system updates the cash of the seller.
- The system updates the stocks of the buyer.
- The system updates the stocks of the seller.
- The system deletes the row of the table “Sell”.

```

<?
session_start();
include("conexion.php");
$buyer = $_GET["buyer"];
$idbuy = $_GET["idbuy"];

$query = "SELECT * FROM User WHERE login = '". $buyer. "'";
$result = mysql_query($query, $conexion);
$row = mysql_fetch_array($result);

$idbuy = $_GET["idbuy"];
$query = "SELECT * FROM Sell WHERE id = ". $idbuy;
$result = mysql_query($query, $conexion);
$row2 = mysql_fetch_array($result);
$amount = $row2["price"] * $row2["volume"] ;
$total = $row["cash"] - $amount;
if ($row["cash"] > $amount){
    //The user that wants to buy will have less cash
    $query = "update User set cash='". $total. " where login='".
$_SESSION['login']. "'";
    mysql_query($query, $conexion);

    //The user that sell will have more cash
    $usertosell = $row2["user"];
    $query = "update User set cash=cash +". $amount. " where id = ".
$usertosell;

```

```

mysql_query($query, $conexion);

//The user that buy will have more stocks
$query = "select * from Ownership where user=".$row["id"]." and
stock=".$row2["stock"]." and price=".$row2["price"];
$result = mysql_query($query, $conexion);
if (mysql_affected_rows($conexion) == 0) {
    $query = "insert into Ownership(user, stock, volume,
price)
values
(".$row["id"].".".$row2["stock"].".".$row2["volume"].".".$row2["price"].")";
mysql_query($query, $conexion);
} else {
    $query = "update Ownership set volume=volume +".
$row2["volume"]." where user=".$row["id"]." and stock=".$row2["stock"];
mysql_query($query, $conexion);
}

//The user that sell will have less stocks
$query = "update Ownership set volume=volume -".
$row2["volume"]." where user=".$usertosell." and stock=".$row2["stock"];
mysql_query($query, $conexion);

//Delete the row with volume=0 in the table ownership
$query = "delete from Ownership where volume <= 0";
$result = mysql_query($query, $conexion);

//Delete the sell in the table
$query="delete from Sell where id=".$idbuy;
mysql_query($query, $conexion);

//Insert the info in History
$query = "insert into History(time, price, buyer, seller, stock,
volume)
values
(now(),".$row2["price"].".".$row["id"].".".$usertosell.".$row2["stock"].".".$row2["volume"].")";
mysql_query($query, $conexion);

header("Location: ../index.php");
}

?>

```

sells2.php

sell2.php is called in **sell.php**. This function checks if someone wants to buy the stocks that the user has put in sell, if this is the case then the sale is conducted automatically. In other case, the system checks if there a row in “Sell” table with the same seller, price and ticker and if so the system updates the volume and in the other case the system inserts a new row in the sell table.

When a sale is made the following actions are executed:

- Delete or update the row in the Buy table, if the buyer wants to buy the same or less stocks with the same price than the seller, the row is deleted. In other case, volume of the stocks is updated.
- The user who buys the stocks will have more stocks. The system checks if the buyer has this ticker in ownership table, if this is the case then the system adds the new stocks to this row in the volume column, if it is not the case the system inserts a new row in the ownership table.
- When the user buys some stocks at a certain price, the cash will automatically be subtracted from the user's account.
- The system inserts or updates the corresponding row of sell table. The system checks if a row exists in the table "Sell" with the same user, price and ticker than this case. If this is the case the system updates the row with more stocks, in other case the system inserts a new row.
- The user who sells the stocks will have fewer stocks. The system subtracts the number of stocks that the seller has in this ticker.
- Delete the row with volume=0 in the table of ownership. The system deletes the rows in ownership table with volume equal 0 because the user leaves, to have this ticker.
- When the user sells some stocks, the system automatically adds the selling amount to the user's account.
- Insert the information in "History" table.

The code is given below:

```

<?
session_start();
include("conexion.php");
$query = "SELECT id AS usersell FROM User WHERE login ='".
$_SESSION['login']."'";
$result = mysql_query($query, $conexion);
$row = mysql_fetch_array($result);

$ownerid = $_POST["ownerid"];
$volume = $_POST["volume"];
$price = $_POST["price"];
$stock = $_POST["stock"];

//check if someone wants to buy this
$query = "SELECT user AS buyer, id AS idbuy, stock AS stock, volume
FROM Buy WHERE stock =".$stock." AND price=".$price;
$result = mysql_query($query, $conexion);
if (mysql_affected_rows($conexion) > 0) {
    $row2 = mysql_fetch_array($result);

    $insertedvolume = $volume;
    if($volume > $row2["volume"])
    {
        $insertedvolume = $row2["volume"];
    }

    //delete or update row of table Buy

```

```

        if($row2["volume"] <= $volume)
        {
            $query = "delete from Buy where id = ".$row2["idbuy"];
            $result = mysql_query($query, $conexion);
        }
        else
        {
            if($row2['volume']-$volume>0){
                $query = "update Buy set volume=volume -".
                $insertedvolume." where id = ".$row2["idbuy"];
                $result = mysql_query($query, $conexion);
                echo "hola";
            }
            else
            {
                $query = "delete from Buy where id = ".
                $row2["idbuy"];
                $result = mysql_query($query, $conexion);
            }
        }

        //The user who buys will have more stocks
        $query = "select * from Ownership where user=".$row2["buyer"]."
        and stock=".$row2["stock"]." and price=".$price;
        $result = mysql_query($query, $conexion);
        if (mysql_affected_rows($conexion) == 0) {
            $query = "insert into Ownership(user, stock, volume,
            price)
            values
            (".$row2["buyer"].",".$row2["stock"].",".
            $insertedvolume.",".$price.)";
            mysql_query($query, $conexion);
        } else {
            $query = "update Ownership set volume=volume +".
            $insertedvolume." where user='".$row2["buyer"]."' and stock=".
            $row2["stock"];
            mysql_query($query, $conexion);
        }

        //The user that wants to buy will have less cash
        $query = "SELECT * FROM User WHERE id = ".$row2["buyer"];
        $result = mysql_query($query, $conexion);
        $rowinfobuyer = mysql_fetch_array($result);
        $amount = $price * $insertedvolume;
        $total = $rowinfobuyer["cash"] - $amount;
        $query = "update User set cash=".$total." where id=".".
        $row2['buyer']."";
        mysql_query($query, $conexion);

        //insert or update row in the table Sell

        $insertedsellvolume = $volume-$insertedvolume;
        $query = "SELECT * FROM Sell WHERE price=".$price." AND stock=".
        $stock." AND user=".$row["usersell"];
        $result = mysql_query($query, $conexion);
        if (mysql_affected_rows($conexion) > 0) {
            $query = "update Sell set volume=volume +".
            $insertedsellvolume." where user=".$row["usersell"]." and stock=".
            $stock." and price=".$price;

```

```

        $result = mysql_query($query, $conexion);
    }
    else{
        if($insertedsellvolume > 0)
        {
            $query = "insert into Sell(price, volume, stock,
user) values (".$price.", ".$insertedsellvolume.", ".$stock.", ".
$row["usersell"].")";
            $result = mysql_query($query, $conexion);
        }
    }

    //The user who sells will have less stocks
    $query = "update Ownership set volume=volume -".
$insertedvolume." where user='".$row["usersell"]." and stock="
.$stock;
    mysql_query($query, $conexion);

    //Delete the row with volume=0 in the table ownership
    $query = "delete from Ownership where volume <= 0";
    $result = mysql_query($query, $conexion);

    //The user who sells will have more cash
    $usertosell = $row["usersell"];
    $query = "update User set cash=cash + ".$amount." where id = ".
$usertosell;
    mysql_query($query, $conexion);

    //Insert the info in History
    $query = "insert into History(time, price, buyer, seller, stock,
volume) values (now(), ".$price.", ".$row2["buyer"].", ".$usertosell.", ".
.$stock.", ".$insertedvolume.)";
    mysql_query($query, $conexion);
}
else{
    $query = "SELECT * FROM Sell WHERE price=".$price." AND stock="
.$stock." AND user=".$row["usersell"];
    $result = mysql_query($query, $conexion);
    if (mysql_affected_rows($conexion) > 0) {
        $query = "update Sell set volume=volume + ".$volume." where
user=".$row["usersell"]." and stock=".$stock." and price=".$price;
        $result = mysql_query($query, $conexion);
    }
    else{
        $query = "insert into Sell(price, volume, stock, user)
values (".$price.", ".$volume.", ".$stock.", ".$row["usersell"].")";
        $result = mysql_query($query, $conexion);
    }
}

header("Location: ../index.php");
?>

```

buyinyourchoice.php

This function is very similar to **sell2.php** but in this case the system automatically checks if there is someone who wants to sell what the current user wants to buy. If it is not the case the function checks if a row exists with the same user, price and ticker, if this is the case the system updates this row if it is not the case the system insert a new row in “Buy” table.

If there is someone who wants to sell this, the function does these actions:

- Delete or update row of table “Sell”. If the buyer wants to buy more or the same amount of stocks that are there in the row under the table “Sell”, this row is deleted, in other case the volume is updated.
- The system updates the stocks of the buyer. The buyer will have more stocks.
- The system updates the stocks of the seller. The seller will have fewer stocks.
- The system updates the cash of the buyer.
- The system updates the cash of the seller.
- Insert the information in History. The sale is saved in the table “History”.

```
<?
session_start();
include("conexion.php");
$query = "SELECT * FROM User WHERE login = '".$_SESSION['login']."'";
$result = mysql_query($query, $conexion);
$row = mysql_fetch_array($result);

$ticker = $_POST["ticker"];
$volume = $_POST["volume"];
$price = $_POST["price"];

$amount = $price * $volume ;
$total = $row["cash"] - $amount;
if ($row["cash"] > $amount){

    //check if someone wants to sell this
    $query = "SELECT user AS seller, id AS idsell, stock AS stock,
volume FROM Sell WHERE stock = ".$ticker." AND price=".$price;
    $result = mysql_query($query, $conexion);
    if (mysql_affected_rows($conexion) > 0) {
        $row2 = mysql_fetch_array($result);

        //delete or update row of table Sell
        if($row2["volume"] <= $volume)
        {
            $query = "delete from Sell where id = ".$row2["idsell"];
            $result = mysql_query($query, $conexion);
        }
        else
        {
            $query = "update Sell set volume=volume -".$volume." where
user=".$row2["seller"]." and stock=".$ticker." and price=".$price;
```

```

        $result = mysql_query($query, $conexion);

    }
    //The user who buys will have more stocks
    $query = "select * from Ownership where user=".$row["id"]." and
stock=".$ticker." and price=".$price;

    $result = mysql_query($query, $conexion);
    $insertedvolume = $volume;
    if($volume > $row2["volume"])
    {
        $insertedvolume = $row2["volume"];
    }
    if (mysql_affected_rows($conexion) == 0) {
        $query = "insert into Ownership(user, stock, volume,
price) values (".$row["id"].",".$ticker.",".$insertedvolume.",".
$price.)";
        mysql_query($query, $conexion);

    } else {
        $query = "update Ownership set volume=volume +".
$insertedvolume." where user=".$row["id"]." and stock=".$ticker." and
price=".$price;
        mysql_query($query, $conexion);

    }

    //The user that wants to buy will have less cash
    $query = "SELECT * FROM User WHERE id = ".$row["id"];
    $result = mysql_query($query, $conexion);
    $rowinfobuyer = mysql_fetch_array($result);
    $amount = $price * $insertedvolume;
    $total = $rowinfobuyer["cash"] - $amount;
    $query = "update User set cash=".$total." where id='".
$row['id'].'.'";
    mysql_query($query, $conexion);

$insertedbuyvolume = $volume - $insertedvolume;
$query = "SELECT * FROM Buy WHERE price=".$price." AND stock=".
$ticker." AND user=".$row["id"];
$result = mysql_query($query, $conexion);
if (mysql_affected_rows($conexion) > 0) {
    $query = "update Buy set volume=volume +".$insertedbuyvolume."
where price=".$price." AND stock=".$ticker." AND user=".$row["id"];
    $result = mysql_query($query, $conexion);
}
else{
    if($insertedbuyvolume > 0)
    {
        $query = "insert into Buy(price, volume, stock, user)
values (".$price.",".$insertedbuyvolume.",".$ticker.",".
$row["id"].")";
        $result = mysql_query($query, $conexion);
    }
}

    //The user who sells will have less stocks
    $query = "update Ownership set volume=volume -".
$row2["volume"]." where user='".$row2["seller"]."' and stock=".
$ticker;

```



```

mysql_query($query, $conexion);

//Delete the row with volume=0 in the table ownership
$query = "delete from Ownership where volume <= 0";
$result = mysql_query($query, $conexion);

//The user who sells will have more cash
$usertosell = $row2["seller"];
$query = "update User set cash=cash + ".$amount." where id = ".$seller;
mysql_query($query, $conexion);

//Insert the info in History
$query = "insert into History(time, price, buyer, seller, stock,
volume) values (now(), ".$price.", ".$row["id"].", ".$seller.", ".$sticker.", ".$insertedvolume.)";
mysql_query($query, $conexion);

}
else{

    $query = "SELECT * FROM Buy WHERE price=".$price." AND stock=".$sticker." AND user=".$row["id"];
    $result = mysql_query($query, $conexion);
    if (mysql_affected_rows($conexion) > 0) {
        $query = "update Buy set volume=volume + ".$volume." where price=".$price." AND stock=".$sticker." AND user=".$row["id"];
        $result = mysql_query($query, $conexion);
    }
    else{
        $query = "insert into Buy(price, volume, stock, user)
values ( ".$price.", ".$volume.", ".$sticker.", ".$row["id"].")";
        $result = mysql_query($query, $conexion);
    }
}

}

header("Location: ../index.php");

?>

```

rss.php

PHP works with RSS besides HTML.

rss.php shows selling and buying of orders. The user who has the RSS is informed of the new rows of 'Buy' and 'Sell' tables, the system shows the ticker, price, volume and date details.

The code is given below:

```

<?php print "<?"; ?>xml version="1.0" encoding="utf-8" <?php print "?
>";
include("conexion.php");
$query = "SELECT * FROM Sell, Stock WHERE Sell.stock = Stock.id";
$answersell = mysql_query($query);
$query = "SELECT * FROM Buy, Stock WHERE Buy.stock = Stock.id";
$answerbuy = mysql_query($query);
$query = "SELECT now() AS time FROM Sell";
$answernow = mysql_query($query);
$rownow = mysql_fetch_array($answernow);
?>
<rss version="2.0">
<channel>
  <title>Virtual Trade</title>
  <link>http://localhost/</link>
  <description>Virtual Trade is a simulation of the real
Trade.</description>
  <language>en-uk</language>
<?
while($row = mysql_fetch_array($answersell)){
?>
<item>
  <title>Sells</title>
  <link>http://localhost/</link>
  <description>
  <?
    echo "Ticker: ".$row['ticker'];
    echo "&lt;br/&gt;";
    echo "Price: ".$row['price'];
    echo "&lt;br/&gt;";
    echo "Volume: ".$row['volume'];
    echo "&lt;br/&gt;";
  ?>
</description>
  <pubDate>
  <?
    echo $rownow["time"];
  ?>
</pubDate>
  <guid>http://localhost/</guid>
</item>
<?
}
while($row = mysql_fetch_array($answerbuy)){
?>
<item>
  <title>Buys</title>
  <link>http://localhost/</link>
  <description>
  <?
    echo "Ticker: ".$row['ticker'];
    echo "&lt;br/&gt;";
    echo "Price: ".$row['price'];
    echo "&lt;br/&gt;";
    echo "Volume: ".$row['volume'];
    echo "&lt;br/&gt;";
  ?>
</description>
  <pubDate>
  <?
    echo $rownow["time"];

```

```

?>
</pubDate>
  <guid>http://localhost/</guid>
</item>
<?
}
?>
</channel>
</rss>

```

graphic.php

graphic.php is called in **details.php**. In **details.php** there is the next JavaScript code, a link and a form to know if the user chooses to see the graphic as per days, per months or per years:

```

<script type="text/javascript">
function check()
{
    number = document.getElementById("link").src.length-1;

    var a = document.getElementById("link").src.substring(0, number)
+ document.getElementById("time").value;
    document.getElementById("link").src = a;
}
</script>
<?

echo "<img src='/graphs/graphic.php?idticker=" . $idticker . "&graph=0'
id='link'>";
?>

<br/><br/>
<SELECT id="time" onChange="check()">
<OPTION value="0" selected>Today</option>
<OPTION value="1">This month</option>
<OPTION value="2">This year</option>
</SELECT>

```

The JavaScript code changes the last character in the link to draw the graphic with the user's selection in the form.

graphic.php generates a graphic based on per days, per months or per years depending of the user's selection. The day's graphic is drawn with the data of the present day. The month's graphic is drawn with the price's average each day in the current month. The year's graphic is drawn with the price's average each month in the current year.

```

<?php
session_start();

```

```

include("../functions/conexion.php");
include ("src/jpgraph.php");
include ("src/jpgraph_line.php");

$idticker = $_GET["idticker"];
$graph = $_GET["graph"];
$ydata = array();
$datax = array();

//if the user selects per days
if($graph == 0){
    $query = "SELECT DISTINCT year(time) AS year, month(time) AS
month, day(time) AS day, hour(time) AS hour, minute(time) AS minute,
year(now()) AS yearnow, month(now()) AS monthnow, day(now()) AS
daynow, time, price FROM History WHERE stock=".$idticker;
    $answer = mysql_query($query);
    while($row = mysql_fetch_array($answer)){
        if ($row["day"] == $row["daynow"]){
            array_push($ydata, floatval($row["price"]));
            $st = $row["hour"].":".$row["minute"];
            array_push($datax, $st);
        }
    }
}
//if the user selects per months
if($graph == 1){
    $query= "SELECT AVG(price), month(time) AS month, day(time) AS
d, month(now()) AS monthnow, time FROM History WHERE stock=".$
$idticker." GROUP BY d";
    $answer = mysql_query($query);
    while($row = mysql_fetch_array($answer)){
        if ($row["month"] == $row["monthnow"]){
            array_push($ydata, floatval($row["AVG(price)"]));
            array_push($datax, $row["d"]);
        }
    }
}
//if the user selects per years
if($graph == 2){
    $query= "SELECT AVG(price), month(time) AS m, year(time) AS
year, month(now()) AS monthnow, year(now()) AS yearnow, time FROM
History WHERE stock=".$idticker." GROUP BY m";
    $answer = mysql_query($query);
    while($row = mysql_fetch_array($answer)){
        if ($row["year"] == $row["yearnow"]){
            array_push($ydata, floatval($row["AVG(price)"]));
            if($row["m"] == 1){
                array_push($datax, "January");
            }
            if($row["m"] == 2){
                array_push($datax, "February");
            }
            if($row["m"] == 3){
                array_push($datax, "March");
            }
            if($row["m"] == 4){
                array_push($datax, "April");
            }
            if($row["m"] == 5){
                array_push($datax, "May");
            }
        }
    }
}

```

```

        if($row["m"] == 6){
            array_push($datax, "Juny");
        }
        if($row["m"] == 7){
            array_push($datax, "July");
        }
        if($row["m"] == 8){
            array_push($datax, "August");
        }
        if($row["m"] == 9){
            array_push($datax, "September");
        }
        if($row["m"] == 10){
            array_push($datax, "October");
        }
        if($row["m"] == 11){
            array_push($datax, "November");
        }
        if($row["m"] == 12){
            array_push($datax, "December");
        }
    }
}

// Create the graph. These two calls are always required
$graph = new Graph(300,200,"auto");
$graph->SetScale("textlin");

// Specify the tick labels
$graph->xaxis->SetTickLabels($datax);

// Create the linear plot
$lineplot=new LinePlot($ydata);
$lineplot->mark->SetType(MARK_UTRIANGLE);

// Add the plot to the graph
$graph->Add($lineplot);

$graph->img->SetMargin(40,20,20,40);
$graph->title->Set("Graphic");
$graph->xaxis->title->Set("Date");
$graph->yaxis->title->Set("Price");

$graph->title->SetFont(FF_FONT1,FS_BOLD);
$graph->yaxis->title->SetFont(FF_FONT1,FS_BOLD);
$graph->xaxis->title->SetFont(FF_FONT1,FS_BOLD);

$lineplot->SetColor("blue");
$lineplot->SetWeight(2);
$graph->yaxis->SetColor("red");
$graph->yaxis->SetWeight(2);
$graph->SetShadow();
// Display the graph
$graph->Stroke();

```

3.5 Software Testing and Test Sets

I tested the Software with artificial data. I introduced users, companies and stocks in the database.

I did a test with the users *Stanley* and *merc* with £1000 of cash, the companies *Barclays*, *Scotland*, *Lloyds* and *Abbey* and the stocks BAR, SCO, LLO and ABB. I put stocks to *merc* and *Stanley* in the “Ownership” table, *merc* has 1 stock of SCO that was sold for £1, 5 stocks of SCO that were sold for £5, 5 stocks of ABB that were sold for £5 and 1 stock of SCO that was sold for £1 and Stanley has 3 stocks of LLO that were sold for £4 each, 2 stocks of SCO that were sold for £5 each, 5 stocks of SCO that were sold for £3 each and 2 stocks of BAR that were sold for £2 each.

merc

Cash:1000

Ticker	Volume	Price
LLO	1	1
SCO	5	5
ABB	5	5
SCO	1	1

Stanley

Cash:1000

Ticker	Volume	Price
LLO	3	4
SCO	2	5
SCO	5	3
BAR	2	2

Stanley wants to sell the two stocks of BAR for £3. And after *merc* wants to buy one stock of BAR for £3.

Therefore, a transaction is made and the data now is:

merc

Cash:1000

Ticker	Volume	Price
LLO	1	1
SCO	5	5
ABB	5	5
SCO	1	1
BAR	1	3

Stanley

Cash:1000

Ticker	Volume	Price
LLO	3	4
SCO	2	5
SCO	5	3
BAR	1	2

4 Critical Appraisal

This project was interesting for me. The trading is not a very familiar area for computer science engineers. I learnt a lot of things searching information about trading and I found that the subject is interesting.

I tried to make the application as realistic as possible and more similar to the trade market but I encountered many difficulties, like the companies not being able to enter an initial price and in the real market there is a lot of statistical data that I do not have.

The project went well as I followed the plan but I made a lot of mistakes at the beginning in PHP programming and SQL queries. I changed the Database and PDF library many times. I would have spent less time in doing my project if I have known this in advance.

I solved the errors introducing the queries in the console of PHPMyAdmin to know if the query is good. I was proving one thing after another in order to figure out what was wrong. I was looking for my errors in various forums in the internet.

If I would have had more time to do the project, I would have looked for more functions in PHP in depth or would have been looking more options in CSS, I could have presented more options to generate PDF files and more graphics.

If at some point I or someone else continues my project, it can be improved in lot many ways if it is added to a forum, more graphics and more options can be added, like displaying the progress of a user or the companies being able to determine the real prices.

I did not have enough time to research deeply about trading and I could not conduct a rigorous test because I had only three months to do the project. A major part of the first two months was taken up by research and approach; programming was largely conducted in the last month. I think the project was well planned. The first week I spent mostly planning and searching for information. The following weeks were spent searching for more information and I also did the Software Requirement Specification, the design and the data base. I was programming PHP after that, I solved a lot of mistakes and at the end I added the option to generate PDF files, graphics and RSS. The last two weeks I wrote the report and I did tests, I found errors and things that I had missed before like when for example you want to buy 2 stocks of the ticker BAR for £2 and there is another user that wants sell 3 stocks of the ticker BAR for £2, the system has to do an automatic sale.

To summarize, I would say that this project was very interesting, I learnt how trading works and I practised programming in PHP and I learnt to solve programming problems in PHP. All in all, working on this project was a very good experience for me.

5 Bibliography and Citations

5.1 Literary Sources

[1] W. R. Page, *Arithmetic out at work: a practical course with examples taken form the retail trade*, J. M. Dent & Sons LTD, 1962.

[2] J.M.W. Tadion, *Deciphering the market: Principles of Chart Reading and Trading Stocks, Commodities and Currencies*, John Wiley & Sons Ltd, Chichester, 1996.

[3] Thomas Connolly and Carolyn Begg, *Database Solutions*, Addison Wesley, London, 2000.

[4] Hakon Wium Lie and Bert Bos: *Cascading Style Sheets*, Addison Wesley, Indiana, 2006.

[5] David Sawyer McFarland: *Dreamweaver MX*, David Pogue, Sebastol, California, 2002.

[6] Ben Laurie and Peter Laurie: *Apache. The Definitive Guide*, O'Reilly, Sebastol, California, 2003.

[7] Luke Welling and Laura Thomson: *PHP and MySQL Web Development*, Developer's library, United States of America, 2004.

[8] Paul DuBois: *MySQL*, New riders, United States of America, 1999.

[9] Gary B. Shelly, Thomas J. Cashman, Steven G.Forsythe and Steven M. Freund: *Mozilla Firefox. Introductory Concepts and Techniques*, Shelly Cashman Series, Boston, 2006.

[10] César Pérez: *Macromedia Dreamweaver MX 2004*, Ra-Ma; Reference, Madrid, 2004.

[11] Diego Montenegro: *Firefox-Thunderbird*, InforBook's S.L., Barcelona, 2006.

5.2 Web Sources

- [12] <http://money.howstuffworks.com/personal-finance/financial-planning/stock.htm>. Homepage of howstuffworks, February 2009.
- [13] <http://uk.finance.yahoo.com/>. Homepage yahoo finance, February 2009.
- [14] <http://www.invertironline.com/Aprender/Nivel4/N4introbolsa1.asp> (Definition and the functions of Trade). Homepage invertironline, February 2009.
- [15] <http://www.uniformserver.com/>. Homepage uniformserver, February 2009.
- [16] <http://php.net/>. Homepage PHP, February 2009.
- [17] <http://www.mysql.com>. Homepage mySQL, February 2009.
- [18] <http://www.w3schools.com/sql/default.asp>. Homepage w3schools, February 2009.
- [19] <http://www.apache.org/>. Homepage apache, February 2009.
- [20] <http://www.aditus.nu/jpgraph/>. Homepage jpgraph, April 2009.
- [21] <http://www.devshed.com/c/a/PHP/PDF-Generation-With-PHP/>. Homepage PDF generation, April 2009.
- [22] <http://www.downes.ca/cgi-bin/page.cgi?post=56>. Homepage downes, April 2009.
- [23] <http://www.fpdf.com>. Homepage FPDF, April 2009.
- [24] <http://wikipedia.org>. Homepage Wikipedia, April 2009.

6 Appendix

6.1 Database with an example of data

Company

Id	Name	Address	Web	Total
1	Barclays	London Road	www.barclays.com	50
2	Scotland Bank	London Road	www.scotlandbank.com	50
3	Lloyds	Abbey Road	www.lloyds.com	1000
5	Abbey	Abbey Road 1	www.abbey.com	100
6	Orange	Welford Road 2	www.orange.com	100
7	Vodafone	Welford Road 1	www.vodafone.com	50
8	O2	Welford Road 2	www.o2.com	100

Stock

Id	Ticker	Company
1	BAR	1
2	SCO	2
3	LLO	3
4	ABB	4
5	ORA	5
6	VOD	6

User

Id	Login	Pass	Name	Address	Age	Cash
1	Merc	Huecija	Mercedes Garcia Martinez	Abbey Road 8	30	867
2	Stanley	Fung	Stanley Fung	Abbey Road 4	30	924
3	User1	Huecija	User1	Abbey Road 8	30	924
4	Laura	Huecija	Laura	Abbey Road 9	20	1000

Ownership

Id	User	Stock	Volume	Price
119	1	2	2	4
120	1	2	5	1
113	2	1	2	5
115	2	1	2	3
114	2	1	2	4
109	1	2	3	5
121	1	1	2	4
107	1	1	2	5

Buy

Id	Price	Volume	Stock	User
52	2	2	3	2
54	4	6	1	1

Sell

Id	Price	Volume	Stock	User
117	5	1	1	1

History

Id	Time	Price	Buyer	Seller	Stock	Volume
9	2009-04-18 17:45:41	5	2	1	3	5
10	2009-04-18 16:53:30	2	2	1	2	0
11	2009-04-19 16:54:53	5	2	1	2	0
8	2009-04-18 17:13:38	5	1	2	3	0
7	2009-04-18 17:12:58	5	2	1	3	0
12	2009-04-19 17:33:18	6	2	1	2	0
13	2009-04-19 18:11:19	5	1	2	3	5

14	2008-04-17 18:11:51	5	2	1	1	5
----	------------------------	---	---	---	---	---

6.2 Queries to create the Database

```

create table Company
(
  id int not null auto_increment primary key,
  name varchar(20),
  address varchar(50),
  web varchar(50),
  total int,
)
create table User
(
  Id int not null auto_increment primary key,
  login varchar(10),
  pass varchar(20),
  name varchar(50),
  address varchar(50),
  age int,
  cash float
)
create table Stock
(
  id int not null auto_increment primary key,
  ticker varchar(20),
  company int,
  foreign key (company) references Company(Id)
)
create table Sell
(
  id int not null auto_increment primary key,
  price float,
  volume int,
  stock int,
  user int,
  foreign key (stock) references Stock(id),
  foreign key (user) references User(id)
)
create table Buy
(
  id int not null auto_increment primary key,
  price float,
  volume int,
  stock int,
  user int,
  foreign key (stock) references Stock(id),

```

```

    foreign key (user) references User(id)
)
create table Ownership
(
    id int not null auto_increment primary key,
    user int,
    stock int,
    volume int,
    foreign key (stock) references Stock(id),
    foreign key (user) references User(id)
)
create table History
(
    id int not null auto_increment primary key,
    time datetime,
    price float,
    buyer int,
    seller int,
    stock int,
    foreign key (buyer) references User(id),
    foreign key (seller) references User(id),
    foreign key (stock) references Stock(id)
)
insert into Company(name, address, web, total) values ("Barclays", "London Road
3","www.barclays.com", 100);
insert into Company(name, address, web, total) values ("Scotland Bank", "London Road
1","www.scotlandbank.com", 100);
insert into Company(name, address, web, total) values ("Lloyds", "Abbey Road 3",
"www.lloyds.com", 100);
insert into Company(name, address, web, total) values ("Abbey", "Abbey Road 1",
"www.abbey.com", 100);
insert into Company(name, address, web, total) values ("Orange", "Welford Road 3",
"www.orange.com", 100);
insert into Company(name, address, web, total) values ("Vodafone", "Welford Road
1","www.vodafone.com", 100);
insert into Company(name, address, web, total) values ("O2", "Welford Road 2",
"www.o2.com", 100);
insert into User(login, pass, name, address, age, cash) values ("merc", "huecija",
"Mercedes Garcia Martinez", 1000);
insert into User(login, pass, name, address, age, cash) values ("stanley", "fung",
"Stanley Fung", 1000);
insert into Stock(ticker, company) values ("ABB", 4);
insert into Stock(ticker, company) values ("ORA", 5);
insert into Stock(ticker, company) values ("VOD", 6);
insert into Stock(ticker, company) values ("O2", 7);
insert into Ownership(user, stock, volume) values (1, 1, 10);
insert into Ownership(user, stock, volume) values (1, 2, 5);
insert into Ownership(user, stock, volume) values (1, 3, 10);
insert into Ownership(user, stock, volume) values (2, 4, 10);
insert into Ownership(user, stock, volume) values (2, 5, 10);

```

```
insert into Ownership(user, stock, volume) values (2, 6, 10);  
insert into Ownership(user, stock, volume) values (2, 1, 5);  
insert into Ownership(user, stock, volume) values (2, 7, 5);
```