StalkStock
Stock Market fantasy Contest

Software Requirements Specification

v1.0

17 January 2019

Parangjothi C
Sankarshan Guru
Pratyush Karmakar

Prepared for
IT350—Software Engineering
Instructor: Biju R Mohan, Ph.D.
Winter 2019

Revision History

Date	Description	Author	Comments
17.01.19	Initial SRS upload	Parangjothi, Pratyush, Sankarshan	Just started SRS
31.01.19	Updated introduction and general description	Parangjothi, Pratyush, Sankarshan	Version 1.1
14.02.19	Updated non-functional requirements	Parangjothi, Pratyush, Sankarshan	Version 1.2
21.02.19	Updated functional requirements and use cases	Parangjothi, Pratyush, Sankarshan	Version 1.3

Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

Signature	Printed Name	Date
Parangjothi C	Parangjothi C	17.01.19
Sankarshan Guru	Sankarshan Guru	17.01.19
Pratyush K	Pratyush Karmakar	17.01.19

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of the Stock Market Fantasy Contest. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the stockholders and the buyers of the system..

1.2 Scope

This software system will be a Stock Market Fantasy Contest for a stockholders and buyers of the stocks. This system will be designed to implement a mock stock market to encourage investors to improve their stock investing skills. More specifically, this system is designed to attract students and investors and entrepreneurs with little or no knowledge to learn more about stocks and better invest in the future in real time stocks. The software will facilitate communication between stockholders and the buyers. The system also contains a relational database containing a list of available stocks and the buyers and sellers of the stocks.

1.3 Definitions, Acronyms, and Abbreviations

1. ASK PRICE: The price that a security holder is willing to sell a security, at a given time.
2. BID PRICE: The price that a buyer is willing to pay for a security.
3. COMMISSION: The fee a broker charges for administering a trade, also known as brokerage fees.ASK PRICE: The price that a security holder is willing to sell a security, at a given time.
4. EXPIRATION DATE: The last day upon which an option or futures contract can be exercised or traded.

This subsection should provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS. This information may be provided by reference to one or more appendixes in the SRS or by reference to other documents.

1.4 Overview

The remaining sections of this document provide a general description, including characteristics of the users of this project, the product's hardware, and the functional and data requirements of the product. General description of the project is discussed in section 2 of this document. Section 3 gives the functional requirements, data requirements and constraints and assumptions made while designing the software product. It also gives the user viewpoint of product. Section 3 also gives the specific requirements of the product. Section 3 also discusses the external interface requirements and gives detailed description of functional requirements. Section 4 is for analysis models for the product.

2. General Description

2.1 Product Perspective

A **stock market**, **equity market** or **share market** is the aggregation of buyers and sellers (a loose network of economic transactions, not a physical facility or discrete entity) of stocks (also called shares), which represent ownership claims on businesses; these may include securities listed on a public stock exchange, as well as stock that is only traded privately. Examples of the latter include shares of private companies which are sold to investors through equity crowdfunding platforms. Stock exchanges list shares of common equity as well as other security types, e.g. corporate bonds and convertible bonds.
StalkStock aims to portray the underlying concept of exchange through an efficient implementation of the current marketplace. The main goal is to provide a platform for investors and brokers for exchange of goods. The platform will serve as a hub to a variety of market trades facilitating to every need of the users.
StalkStock derives its data from existing Stock Market APIs. This data would be used to provide a complete description of the market at the current time to facilitate pricing and efficient exchanges.

2.2 Product Functions

StalkStock is a web based applications which will have 3 main interfaces

1) A Login interface : A login page for the user of the application.
2) An Exchange interface : An interface for users to buy or sell stocks.
3) A Result interface : An interface where users can check their progress.

2.3 User Characteristics

The Buyers or Investors is expected to be Internet literate and be able to buy stocks in the mock stock market.

The Sellers are expected to be Internet literate and to be able to update and display their stocks on the mock stock market.

2.4 General Constraints

WordPress is a free and open-source content management system (CMS) based on PHP & MySQL. Features include a plugin architecture and a template system.There are several APIs available to make plugin development easy.

2.5 Assumptions and Dependencies

The application is web based. Therefore the user is expected to have a browser with flash.

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The interface will provide an interactive design of a stock market fantasy league. The page will be effectively designed to provide users a platform to exchange stocks to gain profits.

3.1.2 Hardware Interfaces

The hardware interfaces involved will include a running personal computer with internet connectivity.

3.1.3 Software Interfaces

The Software interface in question will be showcased as an interactive website.

3.1.4 Communications Interfaces

Communication interfaces include keyboard and mouse.

3.2 Functional Requirements

3.2.1 Login To the Application

3.2.1.1 Introduction

User requires a login to identify themselves.

3.2.1.2 Inputs

1) User name / Email ID
2) Password

3.2.1.3 Processing

Given username or email ID will be used to identify the user. If given correctly (matches a registered user) password matching is done. If password matches they are taken to the home page.

3.2.1.4 Outputs

If Username and password match, they are taken to the home page with their progress.
If it does not match, they are given an appropriate error message.

3.2.1.5 Error Handling

Validate login info

3.2.2 Registration

3.2.2.1 Introduction

New Users require registration to access the website.

3.2.2.2 Inputs

1) User name / Email ID
2) Mobile Number
3) Password
4) Confirm Password

3.2.2.3 Processing

A new user with given username is created and validated. If no other user have the same username and the confirmed password matches, they are taken to the home page.

3.2.2.4 Outputs

If username and password are valid, they are taken to the home page.
If not, they are given an appropriate error message.

3.2.2.5 Error Handling

Validate registration information

3.2.3 Logout

3.2.3.1 Introduction

Logged in users need to logout to finish their session

3.2.3.2 Inputs

A click on the logout button

3.2.3.3 Processing

User will be logged out.

3.2.3.4 Outputs

User is taken back to home page without any progress.

3.2.4 Buy Stock

3.2.4.1 Introduction

Users buy certain number of stocks of a company of their choice.

3.2.4.2 Inputs

1) Company
2) Number of stocks

3.2.4.3 Processing

Given number of stocks of company are added to the user's portfolio and required cash is removed from their account.

3.2.4.4 Outputs

An alert saying their transaction was successful is displayed

3.2.4.5 Error Handling

If the number of stocks makes the price go above the users current money, they are prevented from going through with the transaction.

3.2.4 Sell Stock

3.2.2.1 Introduction

Users sell certain number of stocks of a company of their choice.

3.2.2.2 Inputs

1) Company
2) Number of stocks

3.2.2.3 Processing

Given number of stocks of company are removed from the user's portfolio and required cash is added to their account.

3.2.2.4 Outputs

An alert saying their transaction was successful is displayed

3.2.2.5 Error Handling

If the number of stocks is higher than the number of stocks of the company they own,they are prevented from going through with the transaction.

3.3 Use Cases

i. Casual Description
USE CASE:1 LOGIN
Once a user hits our URL, he will be prompted for his login details-username and password which are then authenticated by our system. This authentication is to make sure every user can access only his account and his trade.

USE CASE 2: REGISTRATION
A new user can join the game at any time by registering his details with the system. The login page has a sign for registration in case he hasn't before.

USE CASE 3: LOGOUT
Once a user finishes playing the game, he gets an option to log out thereby making sure nobody else can access his trade on that particular computer.

USE CASE 4: EDIT MY PROFILE
Once logged in, a user will be able to access his profile and make changes to it. He can even delete his profile if he doesn't need it anymore. Once a profile is deleted, the respective user information will be stored in a separate table for security reasons

USE CASE 5: VIEW TRADING HISTORY
This process with provide the user the ability to look at his past trade transactions. The My Trading History tab will provide a complete list of all the transactions till date. The user can also enter any comments he wishes to make about a trade.

USE CASE 6: VIEW LEADERBOARD
Players are ranked weekly depending on their portfolio values at the end of the week. To enter the ranking, a player's portfolio value must be greater than the average portfolio values of all the users. Also, he should have a minimum of fifteen completed transactions in his history. These decisions were taken to ensure fair play. Winner of each week is given 250$ incentive as virtual money.

USE CASE 7: MARKET ORDER: BUY
This process involves buying stocks using Market Order. The user will be provided with a drop down box which has the names of the stocks offered and their ticker symbols. When the user decides on a stock, the stock information like its buy price, sell price etc are displayed. The user can then purchase any number of stocks he can afford to. Confirmation is displayed and an email notification is sent once the transaction has been made.

USE CASE 8: MARKET ORDER: SELL
This process involves buying stocks using Market Order. The user will be provided with a drop down box which has the names of the stocks offered and their ticker symbols. When the user decides on a stock, the stock information like its buy price, sell price etc are displayed. The user can then purchase any number of stocks he can afford to. Confirmation is displayed and an

email notification is sent once the transaction has been made.

USE CASE 9: LIMIT ORDER: BUY
This process involves buying stocks using Limit Order. The user will be provided with a drop down box which has the names of the stocks offered and their ticker symbols. When the user decides on a stock, the stock information like its buy price, sell price etc are displayed. The user can enter a limit threshold value below which automatic purchase of the stock will be made by the system and an email notification will be sent informing the user about the transaction. If the user doesn't have sufficient funds at this point, an email notification is sent saying that he doesn't have enough funds and the order is deleted from the table.

USE CASE 10: . LIMIT ORDER: SELL
This process involves selling stocks using Limit Order. The user will be provided with a drop down box which has the names of the stocks offered and their ticker symbols. When the user decides on a stock, the stock information like its buy price, sell price etc are displayed. The user can enter a limit threshold value above which automatic selling of the stock will be made by the system and an email notification will be sent informing the user about the transaction. If the user doesn't have sufficient stocks at this point, an email notification is sent saying that he doesn't have enough stocks and the order is deleted from the table.

USE CASE 11: STOP ORDER: BUY
This process involves buying stocks using Stop Order. The user will be provided with a drop down box which has the names of the stocks offered and their ticker symbols. When the user decides on a stock, the stock information like its buy price, sell price etc are displayed. The user can enter a stop threshold value above which automatic purchase of the stock will be made by 15the system and an email notification will be sent informing the user about the transaction. If the user doesn't have sufficient funds at this point, an email notification is sent saying that he doesn't have enough funds and the order is deleted from the table.

USE CASE 12: STOP ORDER: SELL
This process involves selling stocks using Stop Order. The user will be provided with a drop down box which has the names of the stocks offered and their ticker symbols. When the user decides on a stock, the stock information like its buy price, sell price etc are displayed. The user can enter a stop threshold value below which automatic selling of the stock will be made by the system and an email notification will be sent informing the user about the transaction. If the user doesn't have sufficient stocks at this point, an email notification is sent saying that he doesn't have enough stocks and the order is deleted from the table.

USE CASE 13: POST IN FORUM
This process involves allowing the user to start a new thread or reply to an existing one in the forums. Each thread in the forum has its own subject and each post has the username who made the comment.

USE CASE 14: VIEW MY GAME PORTFOLIO
This process provides the user the ability to look at his current portfolio- the stocks he has in it,

the current value of the portfolio as well as his account balance.

USE CASE 15: ADD TO WATCHLIST
This process allows the user to watch his favorite stock prices 24X7. The user can add and delete companies to his watchlist and the system displays latest stock information about those companies every time the user refreshes the page.

USE CASE 16: VIEW TOP FIVE GAINERS/LOSERS
Using this process, the user can see a list of top five price-gaining and losing companies from our current database of 20 companies. These are the top five companies in the database that have the highest and lowest percentage change in the stock price. The percent change is calculated using the yesterday's closing price and the price at which the latest trade has been made.

USE CASE 17: VIEW MOST ACTIVE COMPANIES
Using this process, the user can see a list of top five active companies which are the companies that have the highest liquidity (volume of stocks) from our current database of 20 companies.

3.4 Classes / Objects

1. User (Investor):
-userid: int
-firstname: String
-lastname: String
-aboutme: String
-email: String
-password: String

2. LoginService:
-home_display:boolean
-email: String
-password: String
+verify_login (in email: String, in password: String): bool{Post-condition: user is authenticated and homepage is displayed. Alternate-scenario: user enters wrong username and password and is requested to enter them again. }

3. SessionManager:
-userid: int
-email: String
+is_valid_session(): bool
+getuserid (in email: String): int

4. UserProfileService:
-display_userdata: boolean

+setfirstname(firstname: String):void
+setlastname(lastname: String):void
+setaboutme(aboutme: text):void
+setemail(email: String):void
+getfirstname(userid: int):String
+getlastname(userid: int):String
+getaboutme(userid: int):text
+getemail(userid: int):text
+getuser():int
+register (in user:User): void
+display_userdata(in userid: int):void{Post-condition: First name, Last name, Age, About-me is extracted from the database and displayed.}
+update_Profile(firstname: String, lastname: String, aboutme:text, email: String):void {Post-condition: Changed fields are updated in the userdata table in the database }
+delete_account(in userid: int):void {Post-condition: All information about the user is deleted in the database}

5. TradeDiary:
-Date: String
-Symbol: String
-Comment: String
-isdisplay: boolean
+insert_comment(in Date: Date, in Symbol: String, in comment: String,userid:int): void{ Post-condition: comment is inserted into the trade_diary table in the database}
+display_comment(userid: int):void{Post-condition: all the comments of the user are extracted from trade_diary table in the database and displayed.}

6. Orders
-orderid: int
-company: String
-tickersymbol: String
-action: String
-Order_type: String
-quantity: int
-transaction_date: datetime
-Price: double
-Commission: double
-userid: int
-status: String

8. LimitOrder:
-threshold1: float

9. StopOrder:
-threshold2: float

10. OrderService:
+insert_order(in order: Order):void
+get_orders(in userid: int): Array{Order}
+get_stockholdings(in userid:int): Array{Company, quantity}
+send_email(in email: String): void
+update_cashbalance (in quantity: int, in Price: double, in commission: double, userid: int ,action: String): void{Post-condition: cashbalance is increased or decreased depending on whether shares are bought or sold and is updated in the portfolio table in the database}
+process_limitorders():void
+process_stoporders():void
+delete_expired():void

11. Timer:
+timer_reset():void

12. TradingHistory:
+extract_history(in userid: int): Order

13. StockData:
-symbol: String
-company: String
-ask: double
-bid: double
-stockchange: double
-stockvolume: double
+update_stockdata(): void{Post-condition: stockdata is extracted from yahoo finance and updated in the stockdata table in the database.}

14. Top5Gainers/Losers:
-isvisible_gainers: Boolean
-isvisible_losers: boolean
+extract_gainers(in stockdata: StockData): Array{company, stockchange}{Post-condition: Top 5 companies based on the change% are extracted from stockdata table and displayed}
+extract_losers(in stockdata: StockData): Array{company, stockchange}{Post-condition: Last 5 companies based on the change% are extracted from stockdata table and displayed}

15. MostActiveCompanies:
-isvisible: boolean
+extract_active(in stockdata: StockData): Array{company, stockvolume}{Post-condition: Top 5 companies based on the volume are extracted from stockdata table and displayed}

16. Portfolio:
-portfolio_display:boolean

-portfolioid: int
-cashbalance: double
-portfoliovalue: double
-portfoliocurrent: double
-userid: int
+getcashbalance(userid:int): double
+ setcashbalance(userid:int,cashbalance: double): double
+getportfoliovalue(userid:int): double
+update_portfoliovalue(in userid: int) {Post-condition: Depending on the stock holdings of the user the portfolio value is updated in the portfolio table.}
+display_portfolio(in userid: int) {Post-condition: Displays the portfoliovalue, cashbalance and stockholdings of the user.}
+calculate_portfoliocurrent():void

17. Thread:
-threadid: int
-subject: text
-threadowner: String
-createdDate: date

18. Post:
-id: int
-post: text
-userid: int
-date_entered- date

19. ThreadService:
+CreateThread():void
+CreatePost(threadid:int):void
+getPost(threadid:int):Post
+deletePost(id:int):void

20. Leaderboard:
-position: int
-userid: int
-username: String
-datemodified: date

21. RankingService:
+rankUsers(datemodified:date): void
+displayRanks(): Leaderboard

22. Watchlist:
-id: int

-symbol: String
-company: String
-action: String
-exchange: String
-changev: float
-dayhigh: float
-daylow: float
-volume: double
-isdeleted: int
-userid: int

23. WatchlistService:
+add(symbol:String): void
+delete(symbol:String): void
+displayWatchlist(): void

3.5 Non-Functional Requirements

- **Functionality**

Homepage will show all the options and links which is easy to operate and for the sake of trader's interest, there will be latest headlines from CNBC. We will also have the most comprehensive stock information which is the same as those in the real world (Yahoo! Finance). Also, Username and password are required when an existing user wants to check his profile or portfolio which he already created. A first-time user who wants to try the game should register for the website first and confirm the email sent by our website. Since the portfolios are stored in a database, they are totally private that there is no way for other players to change information. Other services are also provided to protect access to certain resources or information. Additionally, we have "my trade diary" area which is a place for users to write something down about their stock trading, their thoughts on those trades and their prediction for the markets.

- **Usability**

The program will take consideration of characteristics such as human factors, aesthetics and consistency in the user interface. All the operations are listed on the homepage which is easy to navigate for most visitors. Also, the website must be attractive and pleasing to catch the eyes. Most important, there will be a set of instructions on how to play the game and some useful quick links related to stock information.

- **Reliability**

If failure occurs between the program and connected website such as Yahoo! Finance or users' transactions cannot go through, our program should be able to recover from these failures. We backup all the information includes users' personal information in a database and updates accordingly so that we can call the data immediately. This also used to make sure the accuracy of system calculations.

- **Performance**

When searching the website, clicking buttons to jump to another links or doing stock trading, we hope that the website is as efficient as possible, which is concerned with characteristics such as throughput, response time, recovery time, start-up time, and shutdown time.

- **Supportability**

To setup a web server, we must consider that the website can be extended and improved in the future. With the data information updates every day, changes must be made to adapt to the new requirements. Also, it is easier for administrators if the changes are not hard to implement and maintain.

- **Others**

The website should have a strict system to protect users' information and portfolios from being tampered by some invaders. The coding and construction of the system should meet the basic required standard and take consideration of its implementation.

3.6 Design Constraints

The following requirements are not minimal, but are recommended for the best server experience:

- Operating System: the user can access to our website using Windows XP/Vista/7.
- Internet-LAN Connection with minimum bandwidth of 56Kbps are needed to connect to
- Yahoo! Finance and CNBC so that users can get the latest information.
- Screen Display-our website will be well presented if users use a computer with a display resolution of 1024x768 or greater.
- System should be able to run PHP, MYSQL, javascript and Apache HTTP server smoothly.

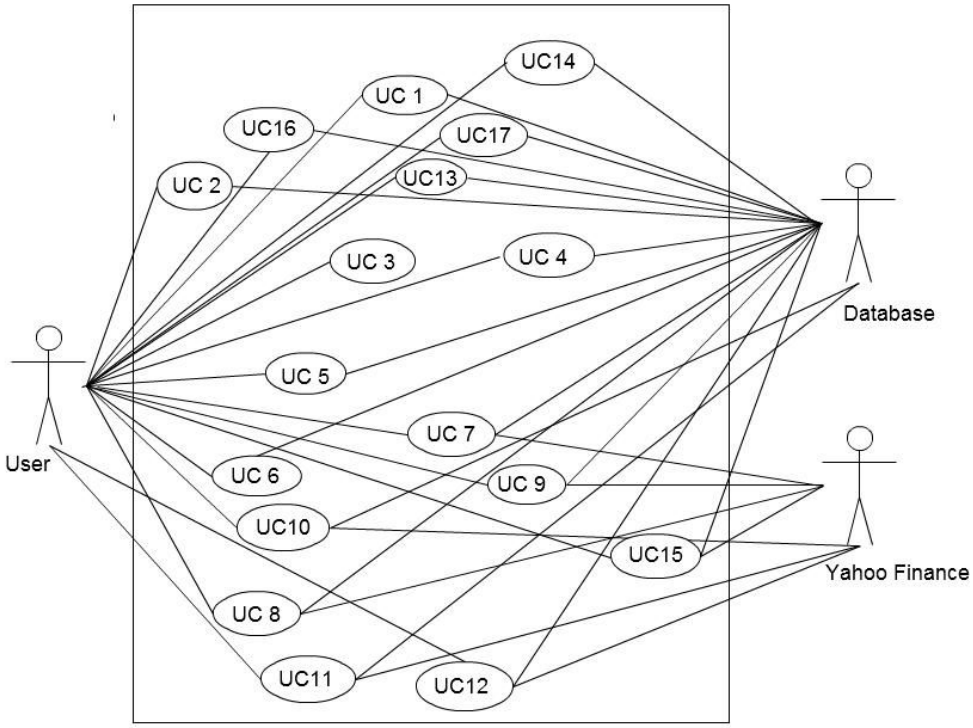
3.7 Logical Database Requirements

The following are the databases that are going to be used:

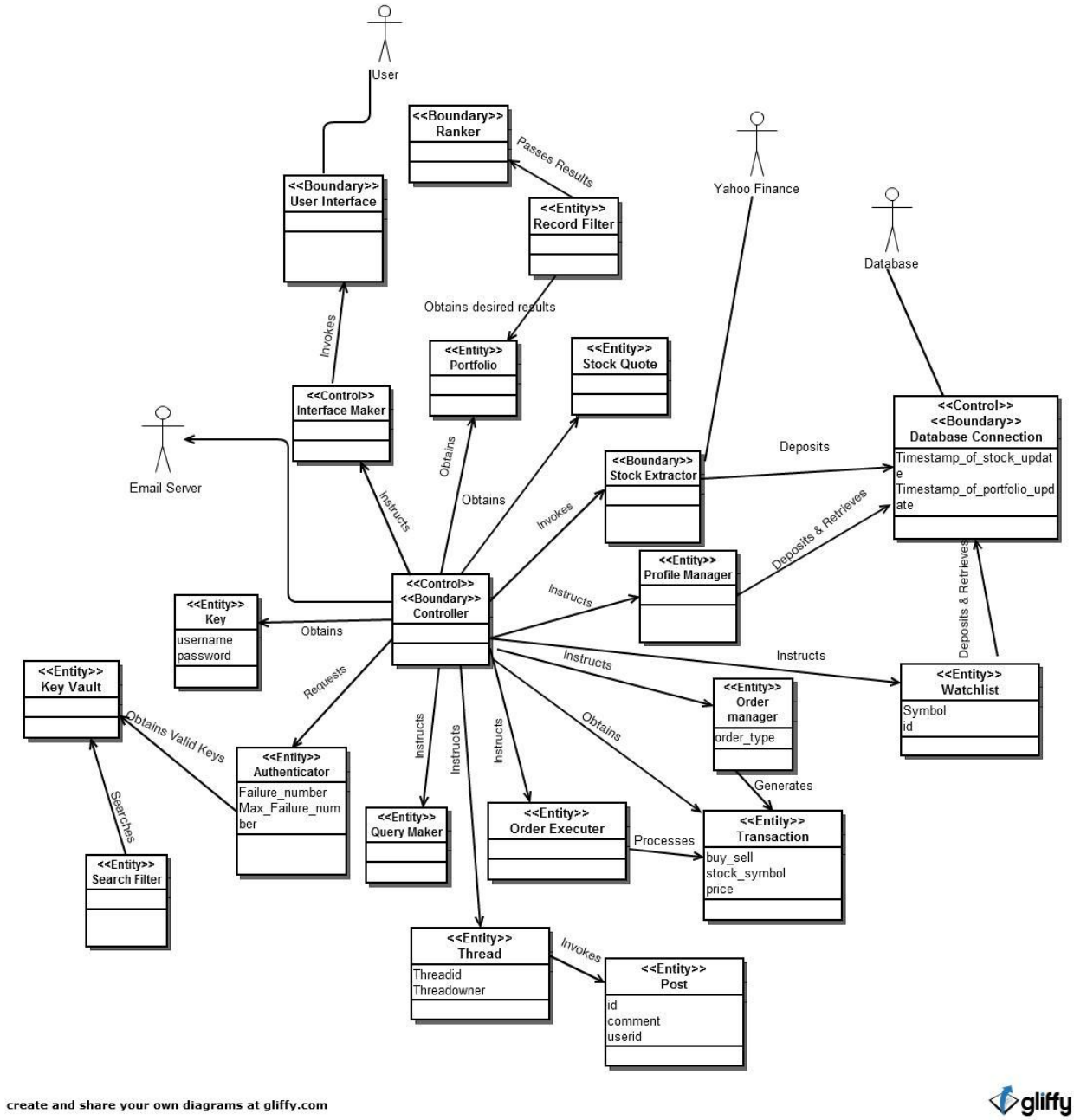
- Database for real-time stock prices
- Database for users registered on the software
- Database to store each user's scores and progress

4. Analysis Models

4.1 Use Case Diagram



4.2 Domain Analysis Diagram



create and share your own diagrams at gliffy.com

gliffy

4.3 Class Diagram