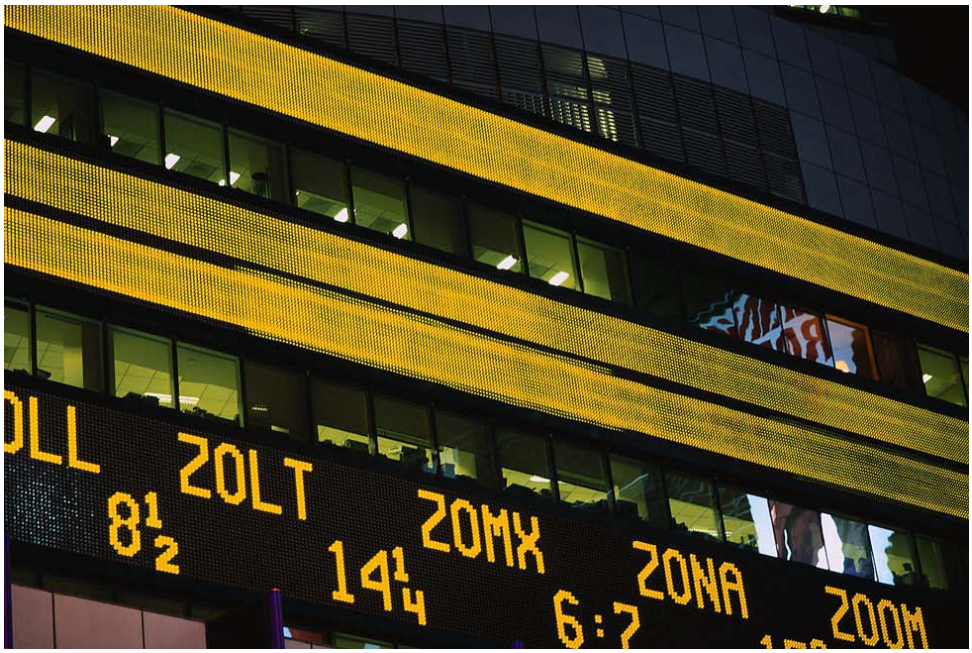
STOCK MARKET INVESTMENT FANTASY LEAGUE - REPORT 1



2/20/2009

URL of the Project Website: http://rahul.rutgers.edu

Rutgers University - 332:452 - Software Engineering

Group #1

Anirban Ghosh, Nikhil Kasthurirangan, Vamsi Kodamasimham, Pramod Kulkarni, and Rahul Sheth

1) CONTRIBUTIONS BREAKDOWN FOR REPORT 1

All team members completed equally to this report, as is evident from the responsibility matrix and allocation chart below.

Name	Anirban Ghosh	Nikhil Kasthurirangan	Vamsi Kodamasimham	Pramod Kulkarni	Rahul Sheth
Report 1					
Project Management	20%	20%	20%	20%	20%
Customer Statement of Requirements		100%			
Glossary of Terms	25%		25%		50%
Functional Requirements Specification	35%	25%	35%	8%	
Nonfunctional Requirements	20%	20%	20%	20%	20%
Domain Analysis	12%		7%	56%	25%
User Interface Design					100%
Plan of Work			100%		
References			100%		

FIGURE 1-1 - RESPONSIBILITY MATRIX

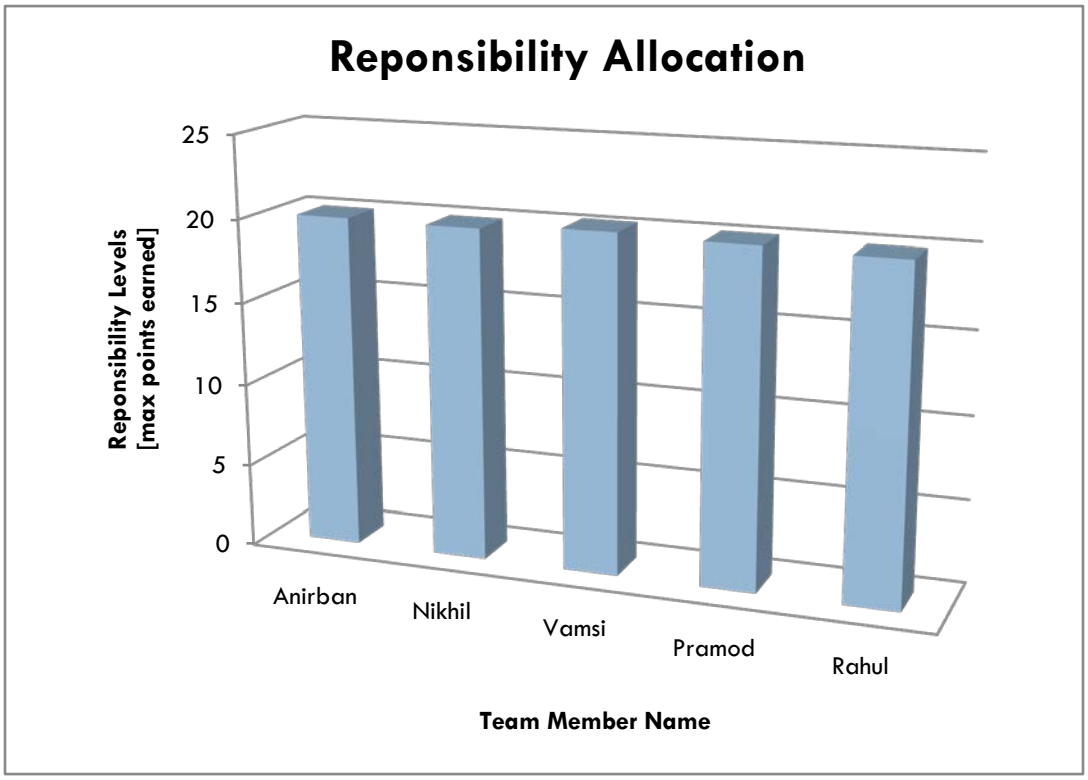


FIGURE 1-1 - RESPONSIBILITY ALLOCATION CHART

Table of Contents

2) CUSTOMER STATEMENT OF REQUIREMENTS

To Whom It May Concern,

Today, I am writing on behalf of demanding investors who are vigorously searching for ways to wisely endow in the stock market while working within thrifty portfolios. To meet this challenge, investors are seeking more effective financial tools that not only motivate them to invest in the stock market, but also provide intelligent strategies to overcome economic fiascos. Currently, one tool that meets these investor requirements is stock market game software. This program simulates a virtual stock world, creating an interwoven network of keen stock investors, through which they trade real-world stocks without the risk of losing real money.

FIGURE 2-1 - EXAMPLES OF CURRENT VIRTUAL STOCK MARKET GAMES [1]

This fiscal tool has given stock investors a way to improve their investing strategies as well as their decision-making skills. Whether the investor is new to the financial markets or a stock investing guru, this is

a powerful mechanism for building skills, for refining his or her strategies, and for gaining important investing experience. For years, stock investors have successfully utilized this tool to enhance their saving and investing techniques. However, due to a recent surge of financial market debacles, the investors are in search of a more sophisticated and influential application that would keep them several steps ahead of economic disasters. Our goal, here at Future Markets, Inc., is to develop such an application. I propose a web-based stock market "fantasy league," where users and investors can trade fantasy stocks using virtual money in a way similar to how they would trade stocks in real markets.

Our aim is for prospective investors to make the most out of our web application, which mirrors the real stock marketplace. To utilize our application, a web user must complete a quick and easy registration process to create his or her private stock portfolio. Then, the registered user (who will be represented as an investor from now onwards) is bestowed with a base amount of virtual money to use to buy stocks. Now, the investors can choose many options to play the game. They can simply join a game that is already in progress, they can create their own games, or they can wait to be invited to a specific game. In any event, the investors are let loose into our virtual stock market game to buy, sell, and modify their stock portfolios as they desire. As the investors trade, our web application should keep track of transactions, update their portfolios, and inform them about these updates. Moreover, our site should conduct real-time, internal updates to regularly determine both the value of an investor's portfolio and the value of each stock in the virtual market. This can be performed by using actual stock data from trusted and reliable external vendors.

Needless to say, our application must provide every investor with options to privately trace his or her stock history. Some sort of personalized interface can be used to achieve this task. This interface must allow an investor to view all the stocks he or she owns, what his or her portfolio's current value is, how much of a gain or loss he or she has made on each stock, and much more. Also, this interface should store past information, as well as the success strategies that top investors have employed. This would keep

investors informed of how they are currently performing, and perhaps remind them to make changes to their portfolios if necessary. Using this historical data, our application should provide smart recommendations on future investment strategies.

Finally, through profits gained from website advertisements, the application administrators should award the best investors on a regular basis. This provides an additional incentive to play the game and perform well. The intent here is for investors to be able to identify practices of the current tournament winner and possibly attempt to emulate his or her styles. On the other hand, the reward may be in the form of stock recommendations such as "players who bought this stock also bought these five others" or even more complex strategies.

Hopefully, the narrative above has given enough information about the demanding requirements. Optimistically, prospective investors who participate in this virtual stock market game will learn more than just how to buy and sell. As they progress further, they will learn core investment concepts and acquire the necessary skills of decision making, saving, and smart investing that will help them succeed not only in the virtual market, but also in real life. After conducting a survey about investors' individual requirements and what they would like to see in a web-based stock market fantasy league application, a majority of responses advised eleven important requirements. I have attached these requirements below.

I wish your team all the best in developing this web application. Please keep me informed of your progress, and let me know if you have any questions.

Sincerely,

Benjamin Stock

Future Markets, Inc.

a) List of Requirements

REQ 1: A stock market fantasy simulator website

After using many stock predictors and stock market simulators, investors are looking for a much more effective and sophisticated financial tool that does it all: provide stock information, analyze, predict, recommend, and finally reward.

REQ 2: Ease of use (with tutorials and guides)

The application must be user-friendly to a point where even a beginner will be able to perform his or her tasks without confusion. If not completely user-friendly, the application must provide a complete user tutorial to introduce these new users to the system.

REQ 3: Reliability

The web site must be reliable with frequent maintenance. More importantly, the application running on the website cannot frequently stall or crash.

REQ 4: Graphical User Interface (GUI)

An user interface with a picturesque layout must be available to perform the stock transactions, and to view stock options and graphs. Ideally, investors must be able to sell or buy stocks with a few mouse clicks.

REQ 5: Separate investor portfolios with privacy

Each investor must be able to register for the application. Once registered, a profile should be created. An investor portfolio must also be maintained, including previous and current stock transactions and amounts of money earned and lost. Also, privacy should be maintained for each investor profile and portfolio.

REQ 6: Ability for multiple users to invest

Several users should be able to register and play with others just like a real stock market.

REQ 7: Ability to automate the buying the selling through thresholds

Users should have the option of setting threshold amounts. So, if the amount drops or rises beyond this threshold, then, the application should either buy or sell according to the set options of the user.

REQ 8: Administrators to monitor activity

A webmaster or an administrator must be available to answer questions and to supervise the game.

REQ 9: Update and evaluate stocks

A continuous update of stocks and portfolios must be provided. These updates must be accurate and reliable to the original stocks present in the stock market.

REQ 10: Recommendations and strategies

In general, through risk analysis and optimization strategies, the application must be able to recommend on what stocks to buy, sell, trade or hold. Also, the application must notify the investors, through email, about recent buying and selling of stocks by other investors.

REQ 11: Leaderboard and rewards

The administrator or the web application itself should determine a winner of the month and award him or her through various ways.

3) GLOSSARY OF TERMS

- **Accessor** - A user interacting with the system that is not already authenticated using the login system
- **Actors** - External entities that the system can interact with
- **Administrator** - A user that is responsible for maintaining and managing the system.
- **AI Mode** - Artificial Intelligence Mode - a mode players can turn on/off that automates their portfolio operations using mathematical models, predictions, and estimates
- **Attributes** - properties of concepts, are usually for storage/accounting purposes or state information
- **Concepts** - Entities that the system is made up of, for intercommunication and handling of events
- **Controller** - Critical system component that manages communication of the system
- **DBMS** - Database Management System - name for a database package (like MySQL or Oracle)
- **FURPS+** - stands for Functionality, Usability, Reliability, Performance, and Reliability (+ Others) - a standard method used to describe and classify non-functional requirements
- **GOMS** - stands for Goals, Operators, Methods, and Selection Rules – a standard method used in creating user interfaces that should outline goals in an easy to read format for use in efficient interface design
- **Graphical Analysis** - Using software to display graphs. This could be of a particular portfolio or a price of a stock over time. It is normally used for comparison as well as analyzing trends of a successful investor
- **Investor** - A user interacting with the system that is authenticated using the login system.
- **Investor Database** - Contains a list of investors currently part of the system and their settings such as user id, passwords, email address, and other personal details
- **Net Worth** - The total value of an investor's portfolio. In this system, it is the sum of current market prices of an investor's stocks
- **Portfolio** - A grouping of financial assets such as stocks, bonds, and cash equivalents, as well as their mutual, exchange-traded, and closed-fund counterparts
- **Share** - The percentage of an industry or market's total sales that is earned by a particular company over a specified time period
- **Stock Database** - Information about the various stocks (such as price quotes, ticker symbols, and market name) are stored in this database
- **Stock Market** - The market in which shares are issued and traded either through exchanges or over-the-counter transactions
- **Timer** - Complex component/actor that synchronizes activities based on timings
- **Use Case** - a use case is a usage scenario of the system, showing an activity a user can perform and the required responses

4) FUNCTIONAL REQUIREMENTS SPECIFICATION

a) Stakeholders

i) Internal Stakeholders

- Owners: A person or people who legally have the right to possess the web application. Owners' interests are profit, cost, performance, expandability, competitors and customer satisfaction.
- Managers: A person who is in charge of the affairs, resources and expenditures of the web application. They interests include performance, growth, customer satisfaction, profit, cost, employees, and demand.
- Employees: A person who maintains the web application by performing maintenance and support. Their interests are reliability, working conditions, salary, working hours, job security, and benefits.

ii) External Stakeholders

- Customers (Users or Investors): A person who registers and creates his private stock portfolio using the web application. The customer's interests include value, quality, reliability and service.
- Advertisers: A company or person who calls the attention of the customer (who uses the web application) to a product or business relating to the stock market or financial industry. Advertiser's interests are number of customers, demand, cost, and competitors.
- Government: The executive policy-making body of the United States. Government's interests are taxation, legislation, unemployment, legality
- Competitors: A company providing the identical products (stock market financial tools) to the general public. Their interests include profit, demand, customers and quality.
- Stock Market: It is a public market where company's shares are traded at an agreed price. Its interests are investors, stock holders, buying and selling techniques, and the economy.
- Stock Researcher: An individual who researches the human behavior in the field of investment in stocks. His or her interests include investors, human behavior, and investing strategies.

b) Actors and Goals

- Accessor: Person who has not been authenticated by the system
  - Type: Initiating
  - Goal: Log In

- Administrator: Person responsible for maintaining and managing the system; also responsible for initial set up of the game
  - Type: Initiating
  - Goal: Initialize the stock market database, maintain the system, monitor users & system usage

- Investor: The person using our system for getting the virtual stock market experience with real-world stock numbers. Each investor has the privilege of viewing other investors' purchased stocks.
  - Type: Initiating
  - Goal: Create investor account, play fantasy game (buying and selling), monitor performance of self, monitor leaderboard

- Advertiser: Person using our system for advertising purposes.
  - Type: Initiating
  - Goal: Post and manager advertisements on the user interfaces

- User: Generalization of the Accessor, Administrator, Investor, and Advertiser

- External Database: The website being used for getting the real-world stock quotes on periodic time intervals.
  - Type: Participating

- Investor Database: Holds portfolio (stock information), portfolio treasury (virtual money) and personal information for each investor.
  - Type: Participating

- Stock Database: Holds the information of all the stocks available in the virtual market.
  - Type: Participating

- Timer: The device primarily responsible for periodically updating the stock database and investor database.
  - Type: Initiating
  - Goal: Inform the system when the Stock market database should be updated, inform the system when the Investor database should be updated

- Email Server: The machine responsible for establishing communication between investors and the system via Email and SMS.
  - Type: Participating

c) Use Cases

i) Casual Descriptions

- **UC 1 - Market Stock Updates:** The Stock Database will be updated on a timely basis. The data will be obtained using an external source. The update frequency will be determined according to the amount of resources available.

- **UC 2 - Investor Stock Updates:** The portfolio in the Investor Database will be updated on a timely basis. The data will be obtained using an external source. The update frequency will be determined according to the amount of resources available.

- **UC 3 - Open an Investor account, create profile:** A user will go through the registration process to gain access to the system.

- **UC 4 - Accessing an existing account:** A user will use the login page to access his or her account.

- **UC 5 - Disabling an existing account:** A user will have the option to disable his or her account. The user's data will be discarded from the investor database.

- **UC 6 - Buy stocks (from market):** User will be able to purchase stocks at market value. User has a choice of quantity.

- **UC 7 - Buy Stocks (from investors):** User will be able to purchase at value set by other investors or make an offer.

- **UC 8 - Sell Stocks (to market):** User will be able to sell stocks at market value. User has a choice of quantity.

- **UC 9 - Sell Stocks (to investors):** User will be able to sell stocks to other investors. User has a choice of quantity and price.

- **UC 10 - Update the Stock Database with new Stocks:** The Stock Database will be updated on a timely basis. The data will be obtained using an external source. The update frequency will be determined according to the amount of resources available. This Use Case is different from UC-1. This Use Case takes care of the updates of Stock Database with newly available stocks in the market.

- **UC 11- Email Notifications:** User will have options to set frequency of email notifications as well as threshold values.

- **UC 12 - Advertisements:** Advertisements will be displayed to generate revenue using online services such as Google AdSense.

- **UC 13 - View history:** A list of previous transactions will be displayed.

- **UC 14 - Graphical Analysis:** Among other graphs, a graph of an investor's net worth is displayed as time progresses.

- **UC 15 - Predictions:** A forecast of an investor's net worth will be displayed according to financial models.

- **UC 16 - Recommendations:** Suggestions such as what stocks to buy and what to sell will be provided.

- **UC 17 - AI Mode:** Users can enable this mode to automate his or her account transactions based on mathematical models.

- **UC 18 - Leaderboard:** A sorted list of investors by his or her net worth, among other metrics.

- **UC 19 - Manage system:** The administrator must manage and maintain the entire system.

ii) Fully Dressed Descriptions

UC 1 – Market Stock Updates:
o Goal: To update the stocks in our virtual market to real-time values. o Initiating Actor: Timer o Participating Actors: External Database, Stock Database. o Pre-condition: Timer countdown is completed; Stock exists in Stock database. o Post-condition: Stock prices are updated and Timer is reset. o Main Success Scenario: • → At configured time intervals, Timer prompts system. • ← System signals for updates. • ← System connects to external database. • ← System requests data from Stock Database. • → Stock Database sends the requested data. • System compares the stock data with external database's stock data. • → External Database returns the updated information. • → System stores the updated information into the Stock Database. • System disconnects with External Database. • ← System signals for Timer to reset. o Alternate Scenario: • ←Access to External Database server cannot be initiated. • System cancels the entire task. • System creates an error log. • ← Timer is reset.

UC 2 – Investor Stock Updates:
o Goal: To update the stocks in Investor's portfolio to real-time values. o Initiating Actor: Timer o Participating Actors: External Database, Investor Database. o Pre-condition: Timer is set; Stock exists in Investor database. o Post-condition: Stock prices are updated and Timer is reset. o Main Success Scenario: • → At configured time intervals, Timer prompts system. • ← System signals for updates. • ← System connects to external database. • ← System requests data from Investor Database.

- → Investor Database sends the requested data.
- System compares the stock data with external database's stock data.
- → External Database returns the updated information.
- → System stores the updated information into the Investor Database.
- System disconnects with External Database.
- ← System signals for Timer to reset.

o Alternate Scenario:
- ← Access to External Database server cannot be initiated.
- System cancels the entire task.
- System creates an error log.
- ← Timer is reset.

UC 3 - Open an Investor account, create profile:

o Goal: To open an investor account and create a profile.
o Initiating Actor: User
o Participating Actors: The Investor Database.
o Pre-condition: The user might have an investor account or the user is new.
o Post-condition: The user will have an investor account or cause of failure will be provided.
o Main Success Scenario:
- → The user supplies User ID.
- ← The System scans the Investor Database for any potential duplicates.
- ← System creates room for the new investor in the Investor Database.
- ← The system asks the investor for more personal information.
- ← The System stores the information in the new entry of the Investor Database

o Alternate Scenario:
- ← The System takes the provided User ID and scans the Investor Database for any potential duplicates.
- ← Presence of duplicate causes System to signal Error to the User.
- ← System requests a different User ID.

UC 6 - Buy stocks (from market):

o Goal: To buy stocks from the market.
o Pre-conditions: The user must be logged into the system.
o Post-conditions: The transaction is reflected in the Investor Database and Stock Database.
Initiating Actor: Investor
o Participating Actors: Investor Database and Stock Database.
o Main Success Scenario:
- → Investor decides to purchase stock by examining the Stock Database.
- ← System retrieves the relevant stock information from the Stock Database.
- ← System attempts to modify the portfolio treasury by the last-updated stock value.
- ← System moves stock information from Stock Database to the portfolio in the Investor Database.
- ← System signals Investor of a successful purchase.

- Alternate Scenario:
  - → Investor decides to purchase stock by examining the Stock Database.
  - ← System retrieves the relevant stock information from the Stock Database.
  - ← System attempts to modify portfolio treasury and finds inadequate funds for making purchase.
  - System cancels entire transaction
  - ← System signals Investor of inadequate funds for the purchase.

UC 7 - Buy Stocks (from investors):

- Goal: Buying stocks from other investors.
- Pre-condition: The user is logged into the system.
- Post-conditions: The transaction is reflected in the portfolios and portfolio treasuries of both Investors.
- Initiating Actor: Investor
- Participating Actors: Investors, Investor Database, and Email Server.
- Main Success Scenario:
  - → An Investor (buyer) decides to buy a stock owned by another investor (seller).
  - ← System retrieves the corresponding stock information from the corresponding portfolio.
  - ← System signals the email server
  - ← Email server receives the stock information and notifies the seller of the offer.
  - → Seller responds with an acceptance of the offer.
  - Email server sends the response to the system.
  - ← System attempts modifications to the portfolio treasuries of both investors.
  - ← System attempts modification to the portfolios of both investors.
  - ← System signals the investor (buyer) of a successful purchase.
  - ← System signals the email server to inform the investor (seller) of a successful sale.
- Alternate Scenario One:
  - → An Investor (buyer) decides to buy a stock owned by another investor (seller).
  - ← System retrieves the corresponding stock information from the corresponding portfolio.
  - ← System signals the email server
  - Email server receives the stock information and notifies the seller of the offer.
  - → Seller responds with a denial of the offer.
  - Email server sends the response to the system.
  - System cancels the transaction
  - ← System notifies the buyer of the purchase denial.
- Alternate Scenario Two:
  - → An Investor (buyer) decides to buy a stock offered by another investor (seller).
  - ← System retrieves the corresponding stock information from the corresponding portfolio.
  - ← System signals the email server
  - Email server receives the stock information and notifies the seller of the offer.
  - → Seller responds with an acceptance of the offer.
  - Email server sends the response to the system.
  - ← System attempts modifications to the portfolio treasuries of both investors.

- ← System finds inadequate funds in the buyer's account.
- System cancels transaction
- ← System notifies buyer of inadequate funds.
- ← System signals Email server to notify seller of an unsuccessful sale due to inadequate funds from the buyer's side.

UC 8 - Sell Stocks (with market):

- Goal: To sell stocks at market price.
- Pre-conditions: The Investor is logged into the system and owns the stock that is to be sold.
- Post-conditions: The transaction is reflected in the portfolio and the portfolio treasury of the Investor.
- Initiating Actor: Investor
- Participating Actors: Investor Database and Stock Database.
- Main Success Scenario:
  - → Investor decides to sell a stock
  - ← The system retrieves data from Investor Database
  - ← The system edits data in stock database
  - ← The system modifies virtual money account with the latest real-time value of the selling stock.
  - ← System signals investor of a successful sale.

UC 9 - Sell Stocks (with investors):

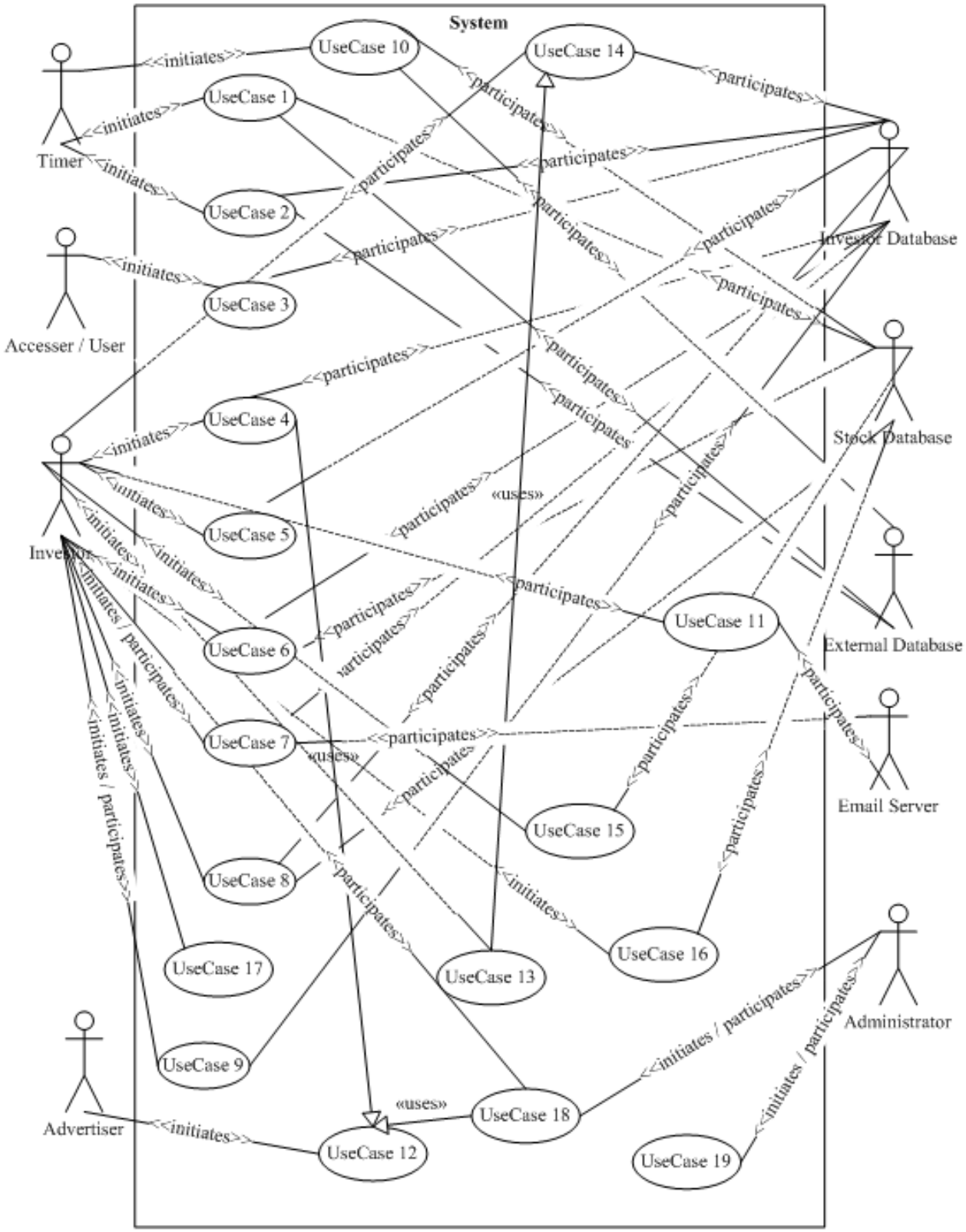
- Goal: To sell stocks to other investors.
- Pre-conditions: The investor is logged in to the system and owns the stock that is to be sold.
- Post-conditions: The transaction is reflected in the portfolios and portfolio treasuries of both investors.
- Initiating Actor: Investor
- Participating Actors: Investors, The Investor Database, and Email Server.
- Main Success Scenario:
  - → An Investor (seller) decides to sell his/her stocks to another investor (buyer).
  - ← System retrieves the corresponding stock information from the corresponding portfolio.
  - ← System signals the email server
  - Email server notifies the buyer of the offer.
  - Buyer responds with an acceptance of the offer.
  - Email server sends the response to the system.
  - ← System attempts modifications to the portfolio treasuries of both investors.
  - ← System attempts modification to the portfolios of both investors.
  - ← System signals the investor (seller) of a successful sale.
  - ← System signals the email server to inform the investor (buyer) of a successful purchase.
- Alternate Scenario (One):
  - → An Investor (seller) decides to sell his/her stocks to another investor (The buyer).
  - ← System retrieves the corresponding stock information from the corresponding portfolio.
  - ← System signals the email server

- Email server notifies the buyer of the offer.
- → Buyer responds with a denial of the offer.
- ← Email server sends the response to the system.
- ← The system cancels the transaction and notifies the seller of a purchase denial.

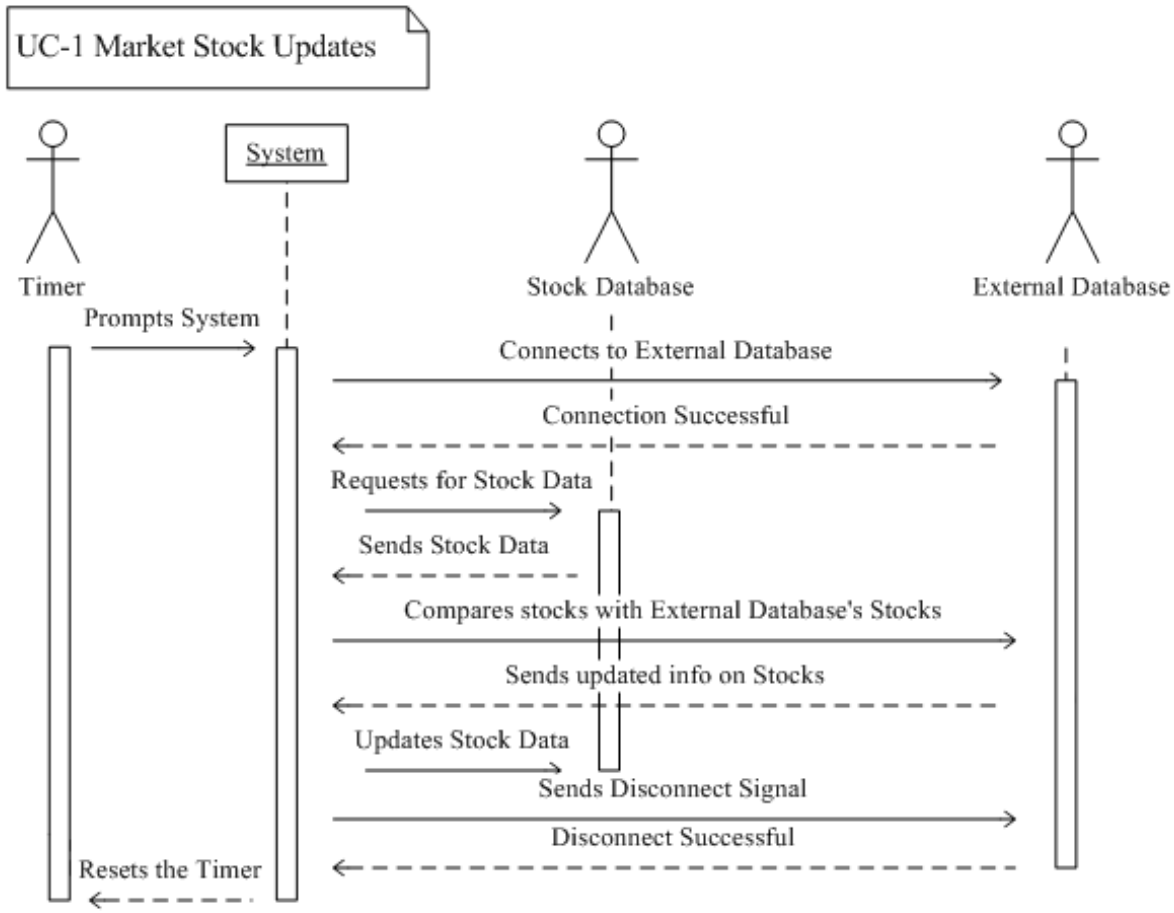
o Alternate Scenario (Two):

- → An Investor (seller) decides to sell his/her stocks to another investor (The buyer).
- ← System retrieves the corresponding stock information from the corresponding portfolio.
- ← System signals the email server
- Email server notifies the buyer of the offer.
- → Buyer responds with an acceptance of the offer.
- Email server sends the response to the system.
- ← System attempts modifications to the portfolio treasuries of both investors.
- ← System finds inadequate funds in the buyer's account.
- ← System cancels transaction and notifies seller of an unsuccessful sale due to inadequate funds from the buyer's side.
- ← System signals Email server to notify buyer of inadequate funds for the purchase.

iii) Use Case Diagram



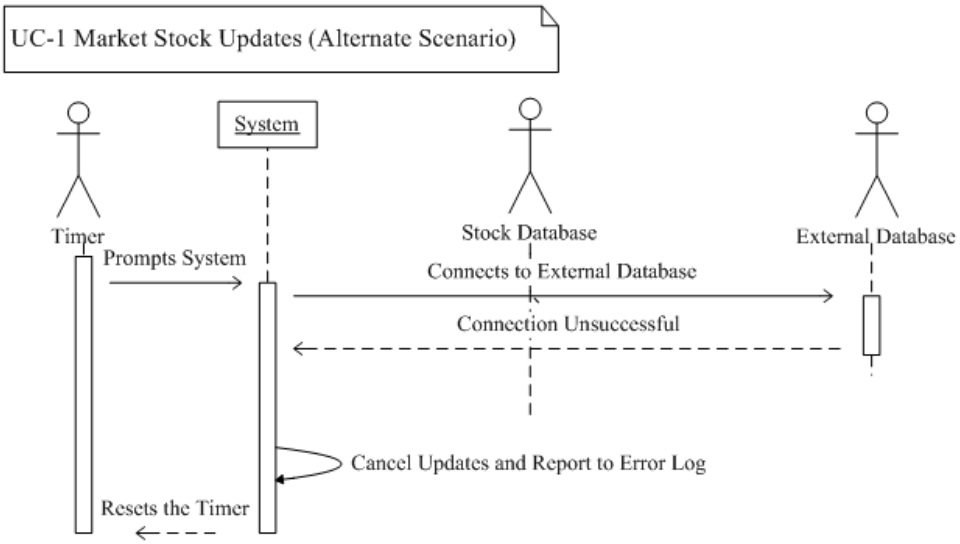
d) System Sequence Diagrams



UC – 1: Market Stock Updates

This figure describes the update of the virtual stock market at regular time intervals (the intervals are not set right now and will be determined in the design). Accurate update of the virtual stock market is crucial to maintain the integrity of the application. So, a trusted external database is utilized to obtain the information of the real world stocks.

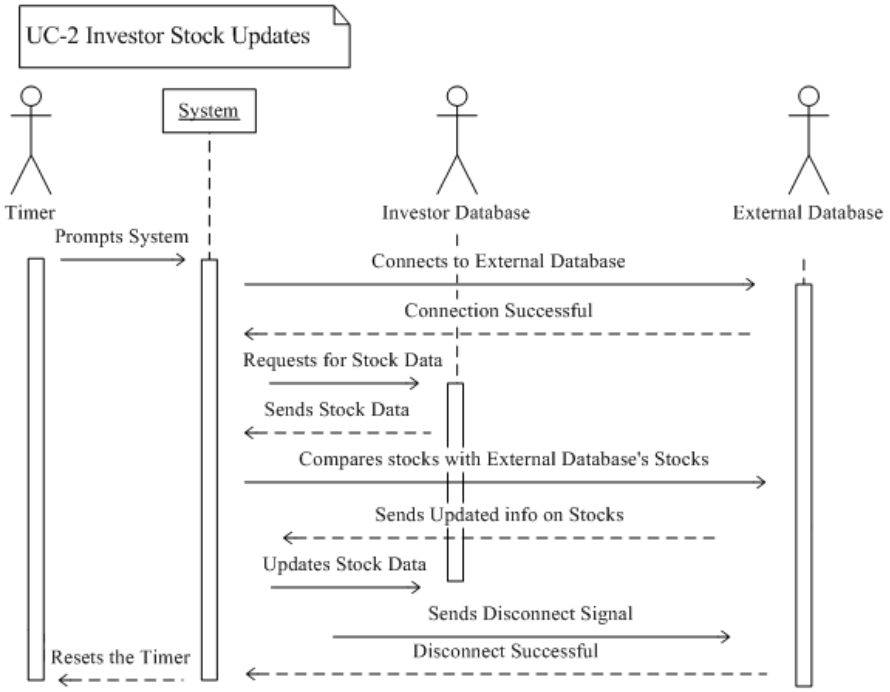
At specific time intervals, the timer prompts the system to initiate the update process. The system attempts connection to the external source. Upon successful connection, it retrieves stock information from the stock database. The information is passed on to the external source, updated and stored back into the stock database. Once the update process completes, the system resets the timer to begin countdown till next update.



UC – 1: Market Stock Updates (Alternate Scenario 1 - Cannot Connect to External Database)

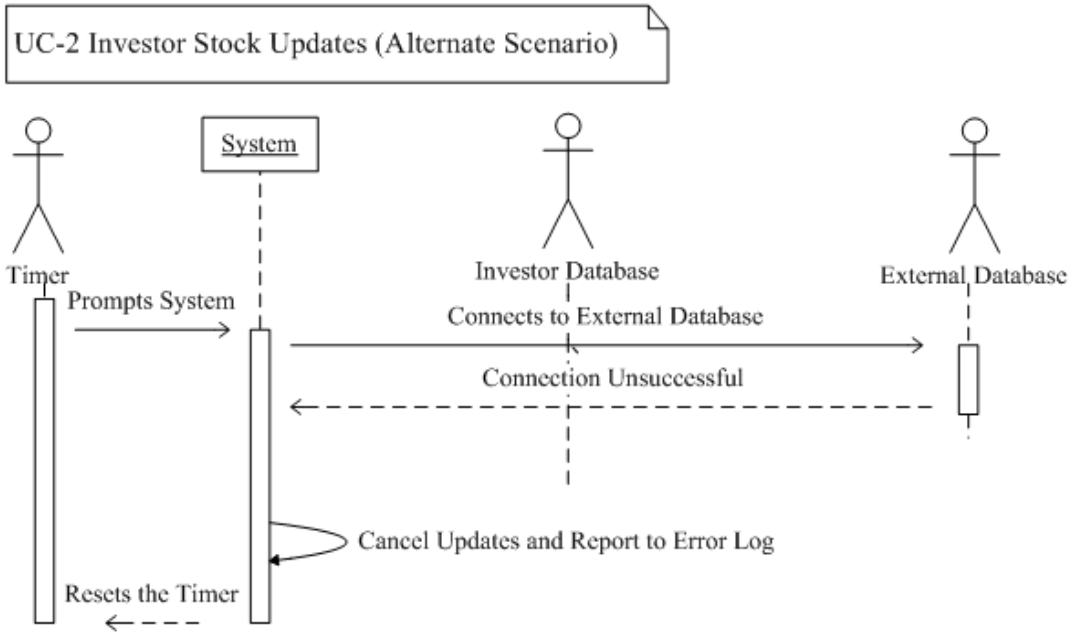
This diagram portrays the situation when events deviate from their prescribed path in the Market Stock Update figure. Such conditions commonly appear in corporate applications. A need for a back-up plan is, therefore, imperative.

At specific time intervals, the timer prompts to system to initiate the update process. The system attempts to connect with the external database. The connection is not established. The system cancels the update and reports this event to an error log. Update retry is initiated by resetting the timer.



UC – 2: Investor Stock Updates

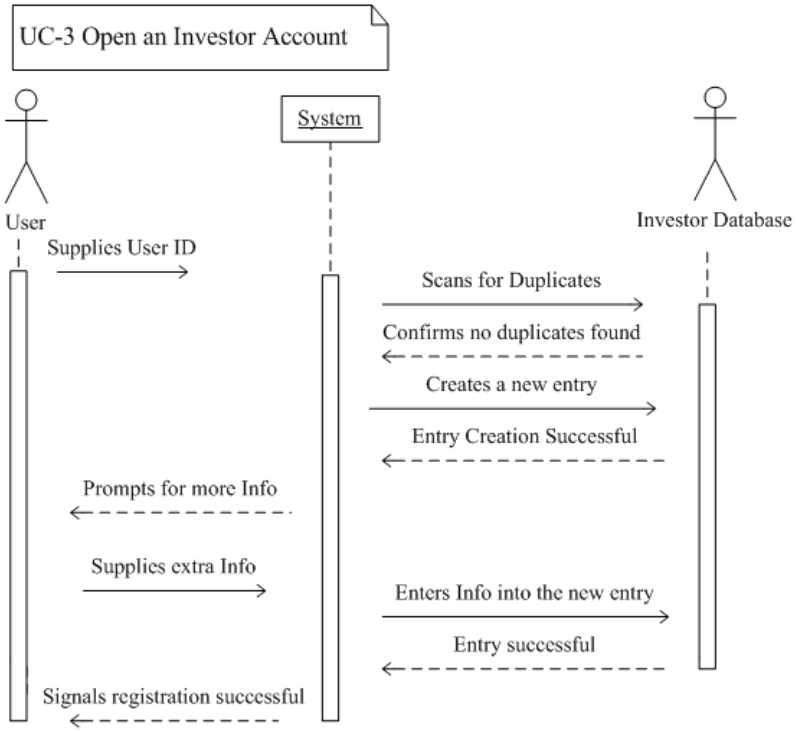
The diagram describes the timely updated of Investor portfolios in the investor database. At specific time intervals, the timer prompts to system to initiate the update process. The system attempts connection to the external database. Upon successful connection, it retrieves stock information from the investor database. Now, the information is passed on to the external database, updated and stored back into the investor database. Once the update process completes, the system resets the timer to begin countdown till next update.



UC – 2: Investor Stock Updates (Alternate Scenario 1 – Cannot Connect to External Database)

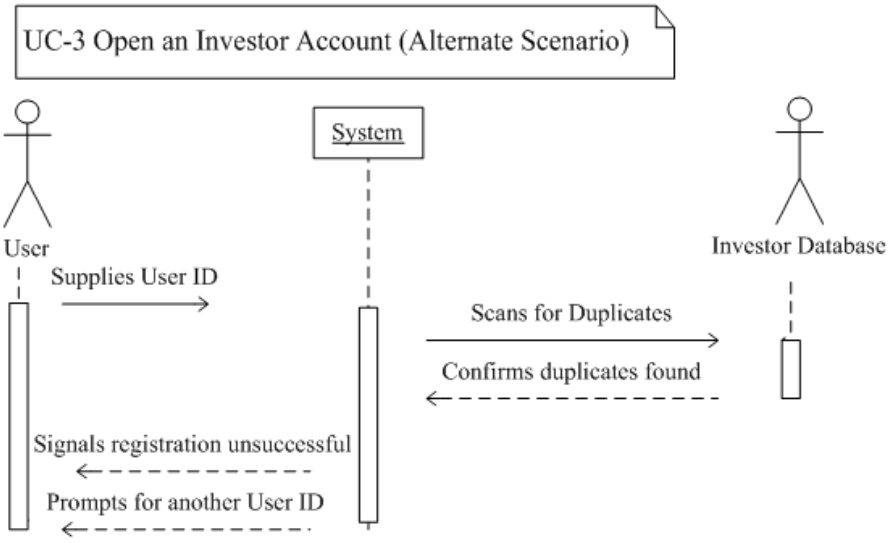
This diagram portrays the situation when events deviate from their prescribed path shown in Investor Stock Update figure.

At specific time intervals, the timer prompts to system to initiate the update process. The system attempts to connect with the external source. The connection is not established. The system consequently cancels the task and reports it to an error log. Update retry is initiated by resetting the timer.



UC – 3: Open an Investor Account

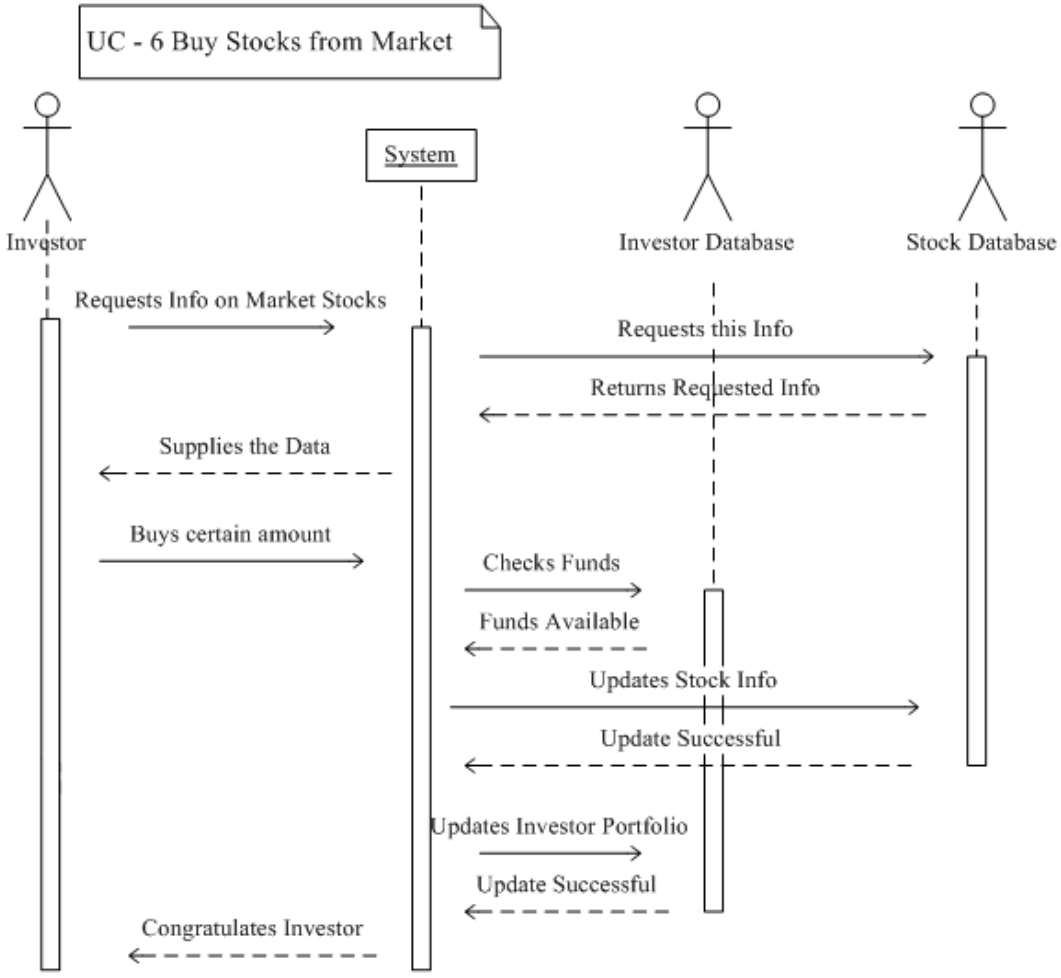
This diagram illustrates a creation of an investor account and portfolio. As an individual begins to register into the application, he or she is asked to provide a User ID. The system takes the provided User ID and scans it against the ID's already present in the investor database. No duplicates are found (main success scenario); the system continues the registration process by asking for some more personal information.



UC – 3: Open an Investor Account (Alternate Scenario - Duplicate User ID)

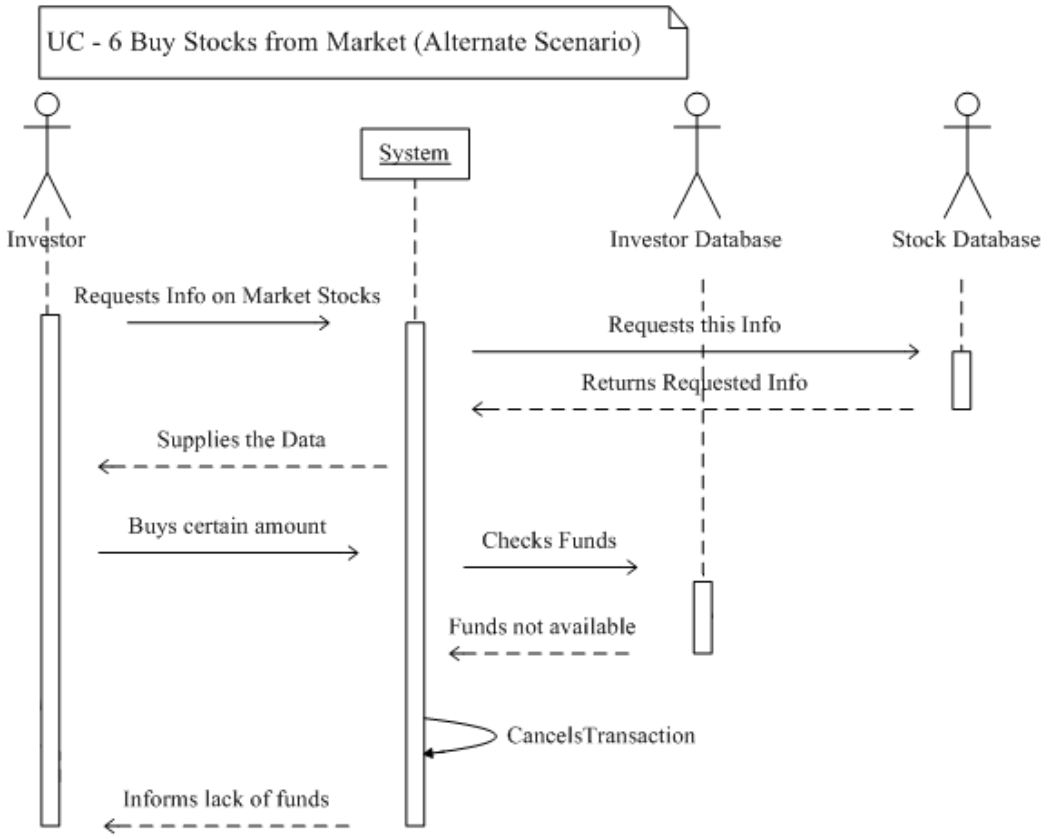
This diagram portrays the situation when events deviate from their prescribed path shown in Open an Investor Account figure.

As an individual begins to register into the application, he or she is asked to provide a User ID. The system takes the provided User ID and scans it against the ID's already present in the investor database. A duplicate is found; the system asks the individual to provide another User ID.



UC – 6: Buy Stocks from Market

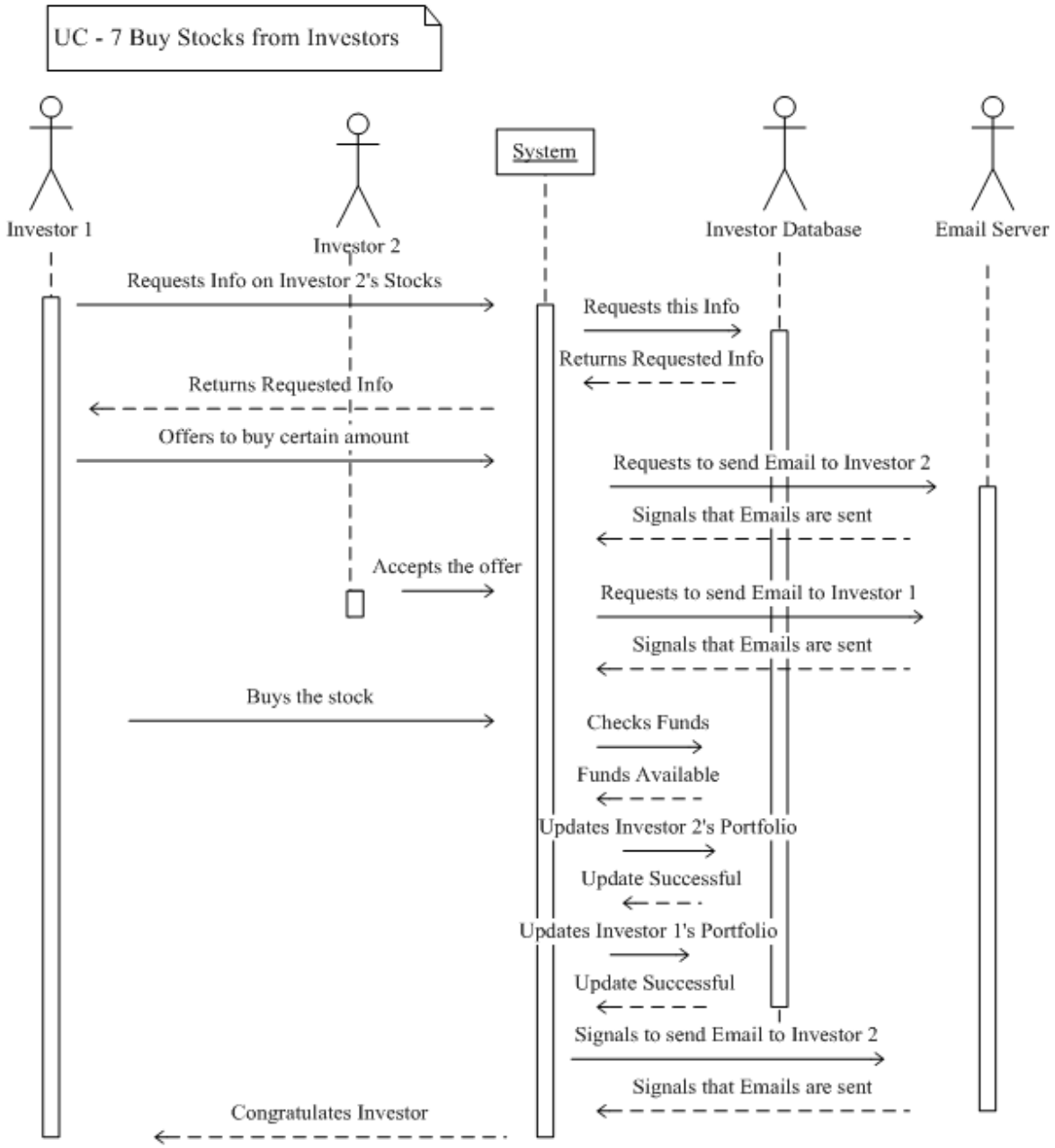
This diagram shows the purchase of a stock from the virtual market by an Investor. The individual requests the particular stock data. The system retrieves the corresponding information from the stock database. When the individual decides to make a purchase, the system checks his or her portfolio treasury for adequate funds. Adequate funds are present (main success scenario). The system makes relevant modifications to the investor portfolio.



UC – 6: Buy Stocks from Market (Alternate Scenario – Lack of Funds)

This diagram portrays the situation when events deviate from their prescribed path show in Buy Stocks from Market figure.

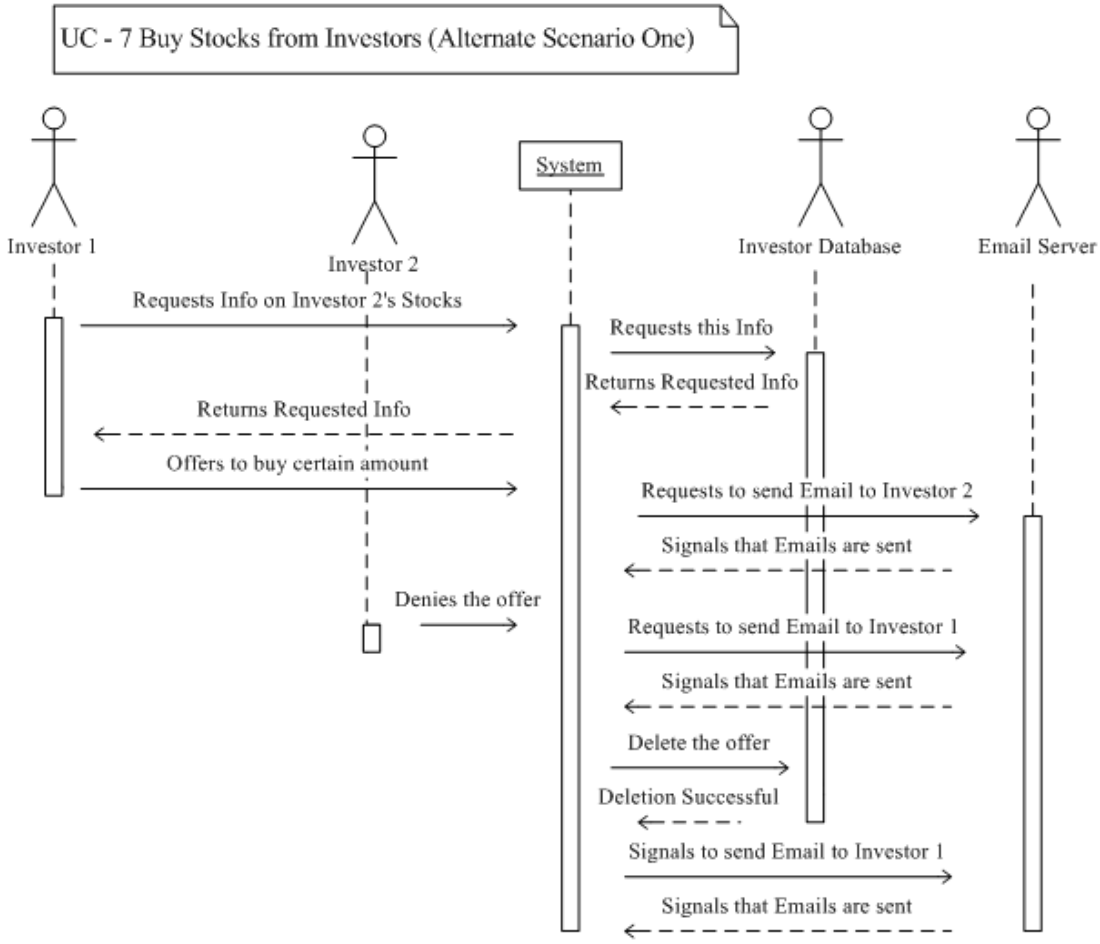
The individual requests stock data and the system retrieves the corresponding information from the stock database. When the individual decides to make a purchase, the system checks his or her portfolio for adequate funds. There is a lack of funds. The system cancels the purchase and notifies the investor.



UC – 7: Buy Stocks from Investors

This picture pertains to an Investor who is buying stocks from other Investors. In order to buy stocks from other investors, the buying investor must be logged into our system (since this is a precondition to this UC, it will not be shown on the diagram). The buying investor scans the investor lists for his or her choice of stock. Once he or she likes a particular stock, he or she sends an offer to the other investor. The system will prompt the e-mail server to send an email about the offer to the other investor.

Then, the other investor accepts the buying investor's offer (this is the main success scenario; in fact, the other investor can deny the offer). The system will prompt the e-mail server to send an email about the acceptance of offer to the buying investor. Now, the buying investor attempts to buy this stock. The system will check whether enough funds are available to make this purchase (this is the main success scenario, so funds are available). The transaction takes place; both investors' portfolios are updated and e-mails are sent out to both investors about the deal.

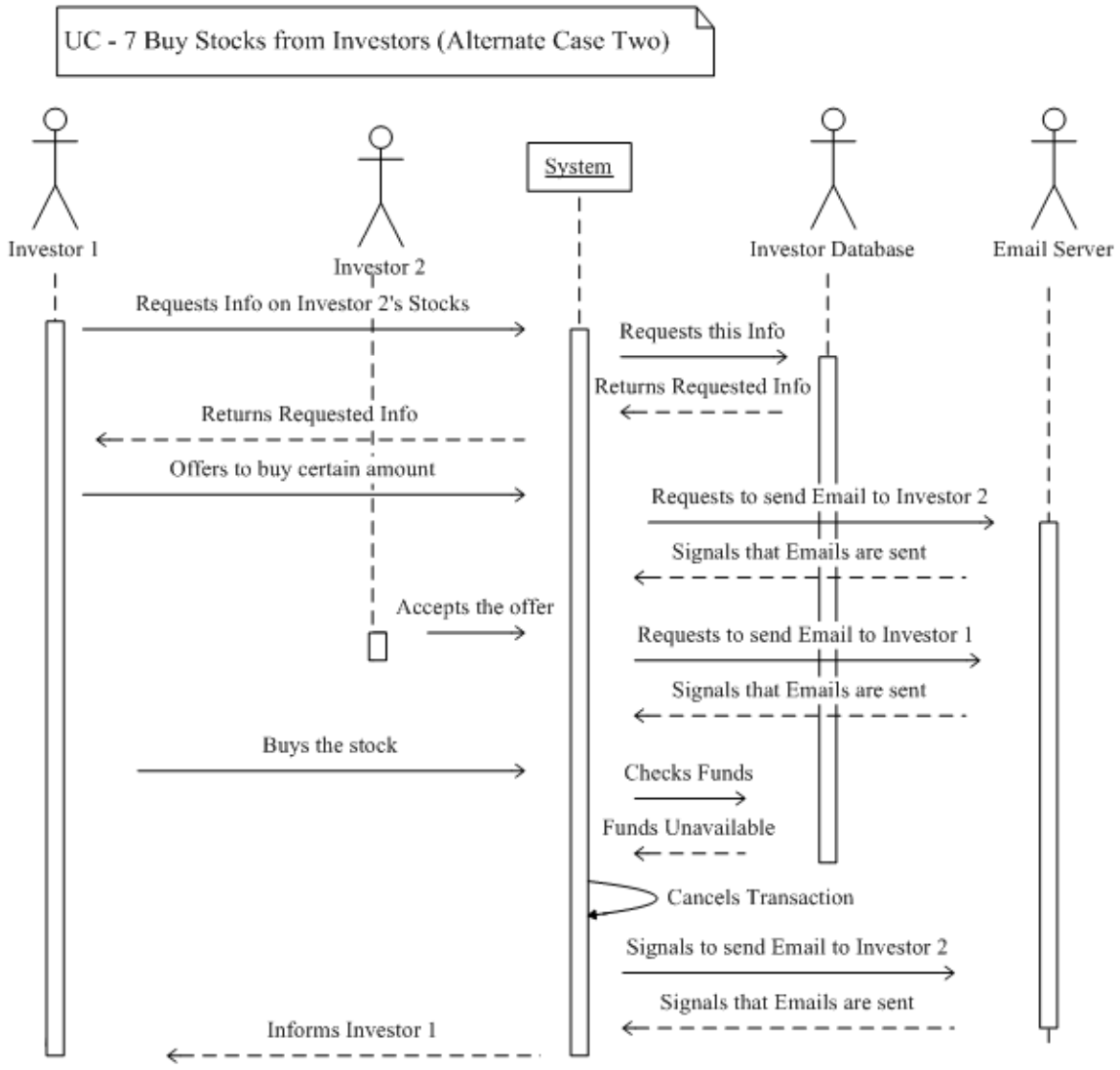


UC – 7: Buy Stocks from Investors (Alternate Scenario 1 – Denial of Offer)

This diagram portrays the situation when events deviate from their prescribed path in the Buy Stocks from Investor figure.

The buying investor scans the investor lists for his or her choice of stock. Once he or she likes a particular stock, he or she sends an offer to the other investor. The system will prompt the e-mail server to send an email about the offer to the other investor.

Then, the other investor denies the buying investor's offer. The system will prompt the e-mail server to send an email about the denial of offer to the buying investor. The transaction is cancelled by the system. The system will prompt the e-mail server to send emails about the cancellation of the deal.

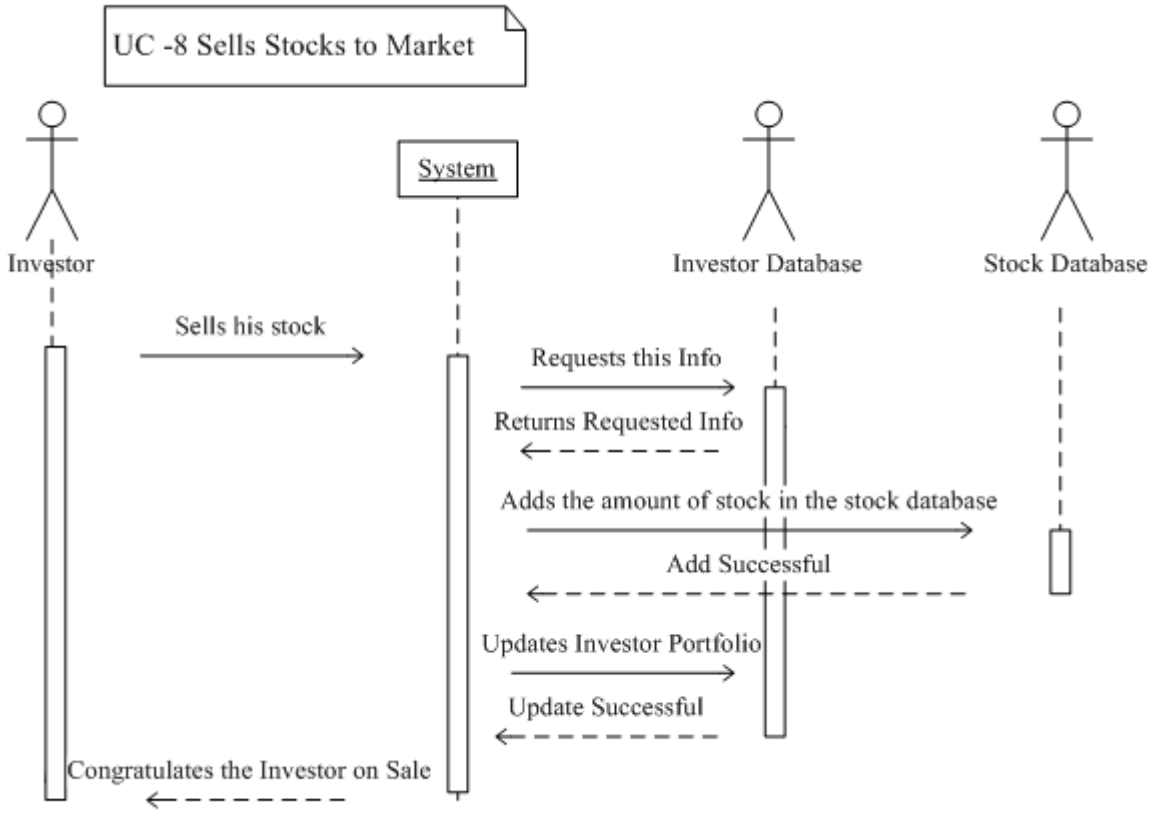


UC – 7: Buy Stocks from Investors (Alternate Scenario 2 – Lack of Funds)

This diagram portrays the situation when events deviate from their prescribed path in the Buy Stocks from Investor figure.

The buying investor scans the investor lists for his or her choice of stock. Once he or she likes a particular stock, he or she sends an offer to the other investor. The system will prompt the e-mail server to send an email about the offer to the other investor.

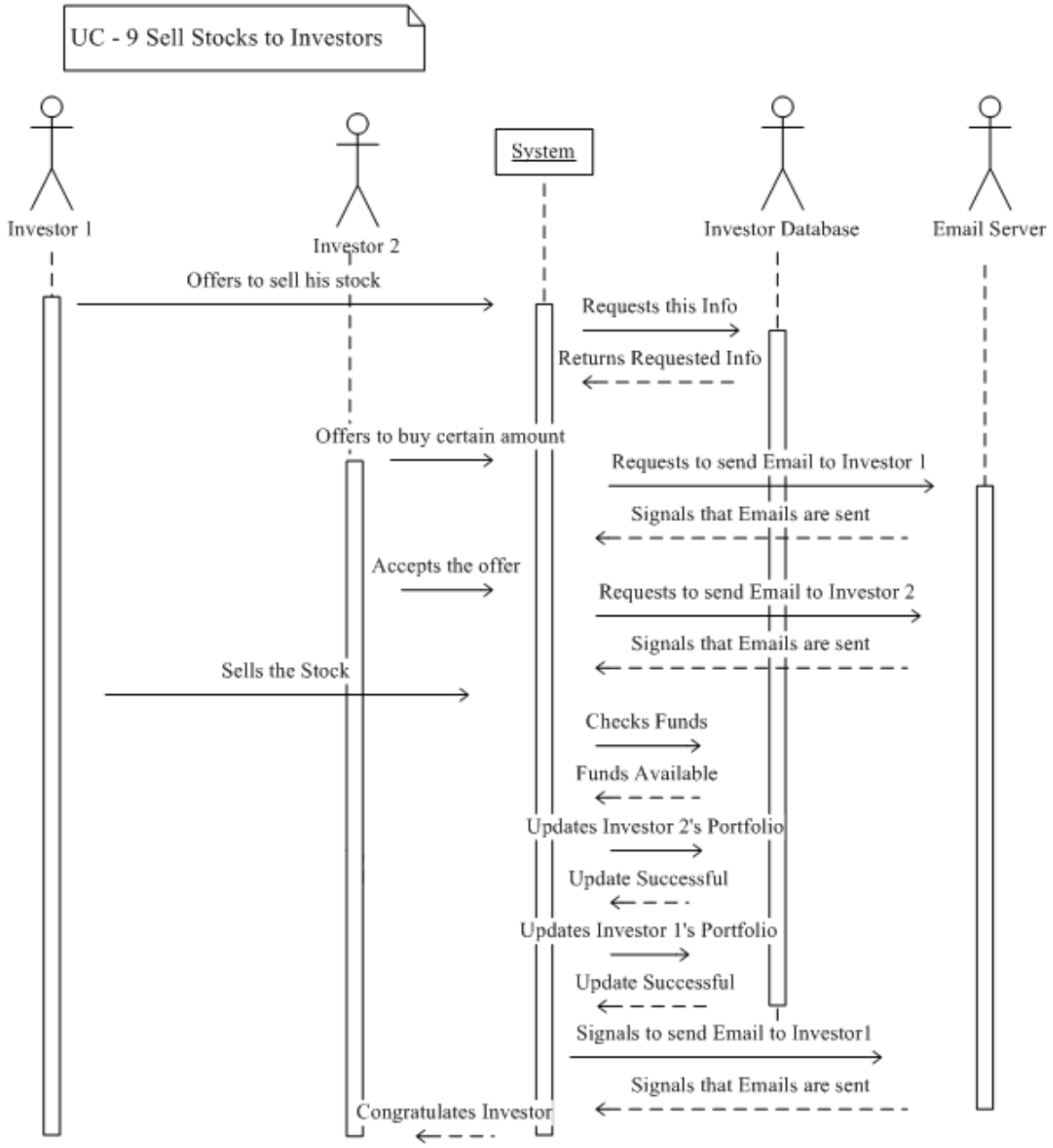
Then, the other investor accepts the buying investor's offer. The system will prompt the e-mail server to send an email about the acceptance of offer to the buying investor. Now, the buying investor attempts to buy this stock. The system will check whether enough funds are available to make this purchase. The system finds that not enough funds are available in the buying investor's stock portfolio. The transaction does not take place and is cancelled by the system. E-mails are sent out to both investors about the cancellation of the deal.



UC – 8: Sell stocks to Market

This figure deals with the selling of stocks, by an investor, to the virtual market. In order to sell stocks to the market, the selling investor must be logged into our system (since this is a precondition to this UC, it will not be shown on the diagram). The selling investor scans his portfolio and decides on a stock that he or she wants to sell. So, after critical decisions, he or she sells the stock.

The system requests this stock information from the investor database. Then, the system updates this stock information in the stock database by adding the shares sold by the investor. After a successful addition, the system updates the investor's portfolio and congratulates him or her on a successful sale.

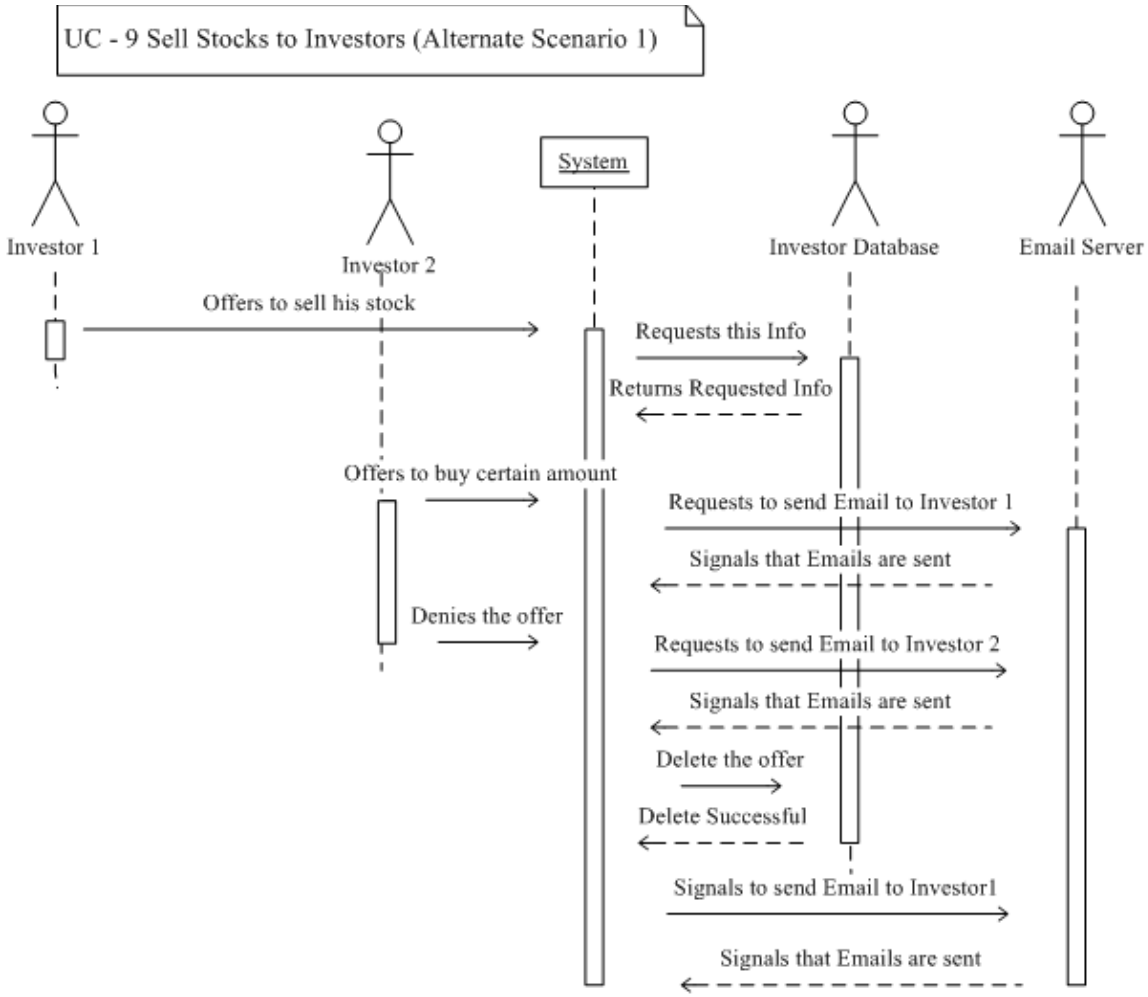


UC – 9: Sell stocks to Investors

This diagram is in regards to the selling of stocks to other investors. In order to sell stocks to other investors, the selling investor must be logged into our system (since this is a precondition to this UC, it will not be shown on the diagram). The selling investor scans his portfolio and decides on a stock that he or she wants to sell. So, after critical decisions, he or she sells the stock.

The system requests this stock information from the investor database. Then, another investor prompts the system by offering to buy the stock. Now, the system signals the e-mail server

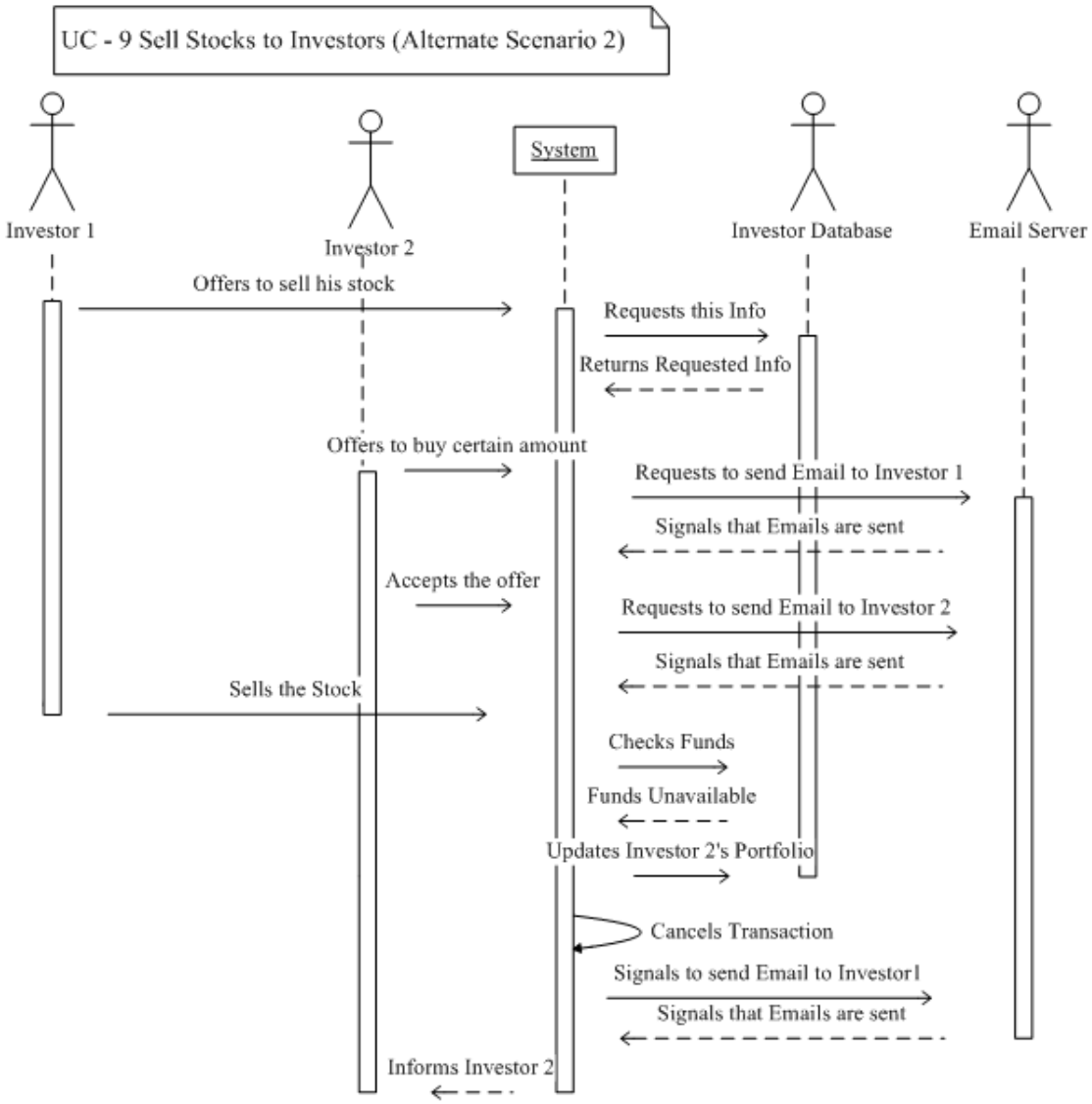
to inform the selling investor about this purchase. The system then checks for enough funds in the buying investor's portfolio (this is the main success scenario, so funds are available). The transaction takes place; both investors' portfolios are updated and e-mails are sent out to both investors about the deal.



UC – 9: Sell stocks to Investors (Alternate Scenario 1 – Denial of Offer)

This diagram portrays the situation when events deviate from their prescribed path in the Sell Stocks to Investors figure.

The selling investor scans his portfolio and decides on a stock that he or she wants to sell. So, after critical decisions, he or she sells the stock. The system requests this stock information from the investor database. Then, another investor prompts the system by offering to buy the stock. Now, the system signals the e-mail server to inform the selling investor about this purchase. However, the buying investor denies this offer of the selling investor. The transaction is cancelled; e-mails are sent out to both investors about the cancellation.



UC – 9: Sell stocks to Investors (Alternate Scenario 2 – Lack of Funds)

This diagram portrays the situation when events deviate from their prescribed path in the Sell Stocks to Investors figure.

The selling investor scans his portfolio and decides on a stock that he or she wants to sell. So, after critical decisions, he or she sells the stock. The system requests this stock information from the investor database. Then, another investor prompts the system by offering to buy the stock. Now, the system signals the e-mail server to inform the selling investor about this purchase.

The system then checks for enough funds in the buying investor's portfolio. However, buying investor's portfolio lacks enough funds to make the purchase. The transaction does not take place and is cancelled. E-mails are sent out to both investors about the cancellation.

5) NON-FUNCTIONAL REQUIREMENTS - *FURPS+*

More information on the FURPS+ model can be found in [2].

Functionality - See Functional Requirements

Usability

The website must be easy to navigate, a delight to use, and pleasing to the eye. This requires a small amount of aesthetic design, as well as forethought into navigation, readability, etc. The user interface should be designed intelligently to meet these criteria.

Reliability

In case of server failure or intrusion, the site should keep a backup of users' data as well as a site-wide snapshot for recovery purposes. These backups should be created routinely and automatically for quick restorations of service. In addition, the data sources for the site should not be limited to one option, in case that source is no longer available (i.e. Yahoo).

Performance

The website should run on a web server without too many hardware demands, as the site itself should be as lightweight and efficient as possible. Users should be able to complete their tasks within a reasonable amount of time, even when many others are logged in/the server is updating quote and prediction information.

Supportability

The site and its server components should have the ability to be extended and improved using modules that either users or administrators can write, but only administrators may install them. This would give the system more flexibility for making future additions and updates. Having a system that can work on many servers to handle a large load may also be a good feature. If this is not possible, perhaps dedicating each of many servers to a specific task (like database management, stock updates, front-end, etc.)

The site should not be coded in a way so that future web servers cannot run it. Thus the site should use modern, standardized languages and protocols to accomplish the tasks defined.

+ (Other)

Each user's portfolio and history should be private. Their login credentials should be stored in a secure manner so that malicious users cannot break into the system and change things as they wish. Login attempts should be monitored and restricted to prevent dictionary/brute-force attacks on the system. IPs should be logged to prevent abuse of account creation tools and multiple accounts. Technologies like "CAPTCHA" should be used to prevent spam and unwanted posts/comments on the site.

6) DOMAIN ANALYSIS

a) Domain Model

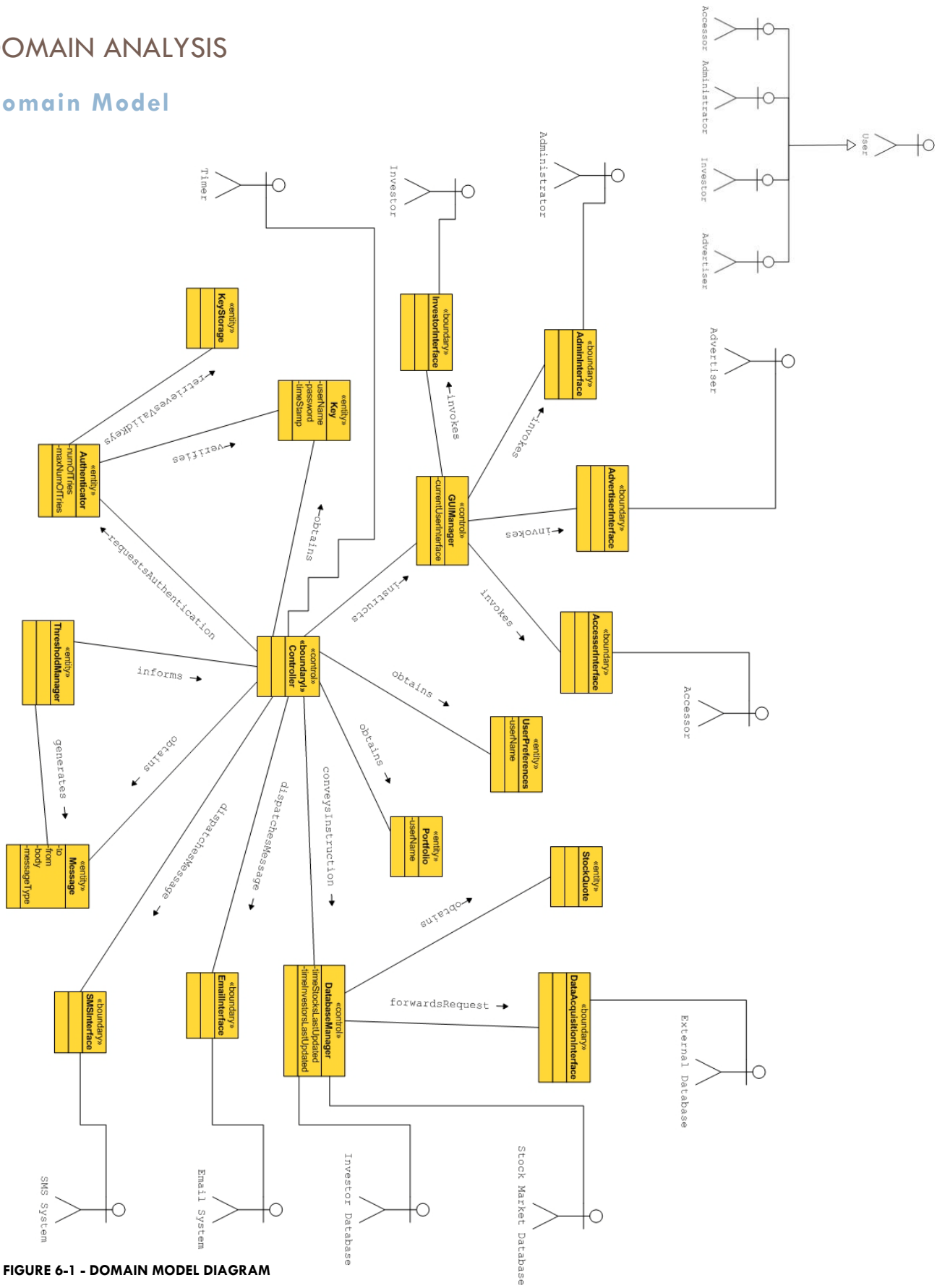


FIGURE 6-1 - DOMAIN MODEL DIAGRAM

DOMAIN MODEL EXPLANATION

Here we have one Controller. The Controller, as defined in the concept table, manages all the functionality required of the system. One peculiarity about the primary Controller is that it is also, in part, a boundary concept. Although our system has one main controller, there are also some subordinate controllers as well, the GUIManager, and DatabaseManager. We decide to use multiple controllers so that some can be specialized. The controller instructs the GUIManager on what interface to present to a user, and the GUIManager conveys all information going between the user and the system. The controller also conveys to the DatabaseManager, all queries and information to be stored. During operations of retrieving and storing data the system manages the data by encapsulating it inside UserPreferences, Portfolio, StockQuote, Key, and Message. The ThresholdManager is the entity that monitors user's thresholds and notifies the primary Controller when a message needs to be sent to an Investor. The Authentication mechanism is directly controlled by the main Controller, as can seen directly from the Domain Model.

i) Concept Definitions

Type = Responsibility Type → "D" = Doing, "K" = Knowing

	Concept Name	Type	Boundary?	Responsibility Description
1	Controller	D	Both	Coordinate actions and information related to a logical subset of the Use Cases
2	GUIManager	D	No	Coordinates with the controller to present the correct user interface
3	AccessorInterface	D	Yes	System interface for the Accessor actor; the interface with which all actors interact before identifying themselves to the system
4	AdminInterface	D	Yes	System interface for the Administrator actor
5	InvestorInterface	D	Yes	System interface for the Investor actor
6	AdvertiserInterface	D	Yes	System interface for the Advertiser actor
7	DataAcquisitionInterface	D	Yes	System interface to the source for stock information; basically the mechanism used for data acquisition from Yahoo Finance
8	DatabaseManager	D	Yes	System interface to the three databases; used for retrieving data from, and storing data in, databases
9	Authenticator	D	No	Uses the User Actor's input to authenticate the User Actor as an Investor, Administrator, etc.
10	Key	K	No	Contain the authentication information inputted by the User Actor
11	KeyStorage	K	No	Internal database for securely storing valid authentication keys
12	EmailInterface	D	Yes	System interface to the Email system
13	SMSInterface	D	Yes	System interface to the SMS system
14	UserPreferences	K	No	Store the preferences of an investor, or administrator
15	ThresholdManager	D	No	Updates user profiles based on their preferences, and current stock data; notifies email & sms systems
16	Portfolio	K	No	Contains a given users portfolio; this is information that will be stored in the Investor Database
17	Message	K	No	Contains message to be dispatched by the controller to the EmailInterface or SMSInterface for sending
18	StockQuote	K	No	Contains information pertaining to one stock quote; this is data that will be stored in the Stock Market Database

TABLE 6-1 - CONCEPT DEFINITION TABLE

ii) Association Definitions

Concept	Relation [direction: →]	Concept
Controller	Instructs	GUIManager
Controller	Obtains	UserPreferences
Controller	Obtains	Portfolio
Controller	conveysInstruction	DatabaseManager
ThresholdManager	informs	Controller
Controller	dispatchesMessage	EmailInterface
Controller	dispatchesMessage	SMSInterface
Controller	requestsAuthentication	Authenticator
Controller	Obtains	Key
Controller	Obtains	Message
Authenticator	Verifies	Key
Authenticator	retrievesValidKeys	KeyStorage
GUIManager	invokes	AccessorInterface
GUIManager	Invokes	InvestorInterface
GUIManager	Invokes	AdvertiserInterface
GUIManager	Invokes	AdminInterface
DatabaseManager	forwardsRequest	DataAcquisitionInterface
DatabaseManager	Obtains	StockQuote
ThresholdManager	Obtains	Message

TABLE 6-2 - ASSOCIATION DEFINITION TABLE

iii) Attribute Definitions

Concept	Attributes
GUIManager	(1) `currentUserInterface`: indicates which user interface is being presented to the user of the system
DatabaseManager	(1) `timeStocksLastUpdated`: time stamp representing when the Stock Database was last updated (2) `timeInvestorsLastUpdated`: time stamp representing when the Investor Database was last updated
Authenticator	(1) `numOfTries`: number of failed login attempts during current session (2) `maxNumTries`: number of failed login attempts before user account is temporarily locked
Key	(1) `userName`: a user's login id (2) `password`: a user's authentication key (3) `timeStamp`: time stamp of current login attempt
UserPreferences	(1) `userName`: a user's login id

Portfolio	(1) `userName`: a user's login id
Message	(1) `to`: target recipient (2) `from`: identifiable system name (3) `body`: contents of the message (4) `interfaceToUse`: indicates whether message is for email, or sms transmissions

TABLE 6-3 - ATTRIBUTE DEFINITION TABLE

b) System Operation Contracts

Operation:	UC 1 – Market Stock Updates
Preconditions:	• controller has received a signal from the Timer to update
Postconditions:	• timeStocksLastUpdated holds time stamp of initiating Timer signal • Stock Database is not empty

Operation:	UC 2 – Investor Stock Updates
Preconditions:	• controller has received a signal from the Timer to update • current time is greater than timeInvestorsLastUpdated
Postconditions:	• timeInvestorsLastUpdated holds time stamp of initiating Timer signal • Investor Database is not empty

Operation:	UC 3 – Open an Investor account, create profile
Preconditions:	• The AccessorInterface is presented to the user
Postconditions:	• If outcome is successful - the InvestorInterface is presented to the user - the Investor Database has a new entry of the new Investor - the KeyStorage has a new entry for the new valid authentication key • If the outcome is not successful - No changes have been made to the system

Operation:	UC 6 – Buy stocks (from market)
Preconditions:	• The InvestorInterface is presented to the user • The investor's portfolio contains sufficient funds
Postconditions:	• Appropriate changes are made to the relevant Investor portfolio, which is stored in the InvestorDatabase

Operation:	UC 7 – Buy stocks (from investors)
Preconditions:	• The InvestorInterface is presented to the user • The investor's portfolio contains sufficient funds
Postconditions:	• Appropriate changes are made to the two relevant Investor portfolios, which are stored in the InvestorDatabase

Operation:	UC 8 – Sell stocks (with market)
Preconditions:	• The InvestorInterface is presented to the user • The investor's portfolio contains the stocks to be sold
Postconditions:	• Appropriate changes are made to the relevant Investor portfolio, which is stored in the InvestorDatabase

Operation:	UC 9 – Sell stocks (with investors)
Preconditions:	• The InvestorInterface is presented to the user • The investor's portfolio contains the stocks to be sold
Postconditions:	• Appropriate changes are made to the two relevant Investor portfolios, which is stored in the InvestorDatabase

c) Mathematical Models

It can be useful for the user to see various calculations based on his or her portfolio. One such value could be VaR, or Value at Risk [5].

The Variance-Covariance method of calculating VaR takes a dataset of daily portfolio returns and assumes they are normally distributed. In a normal curve, the standard deviation is a known statistic that determines how likely certain events are to occur. For example, if an event is ten standard deviations "away" from the mean, it will likely never happen.

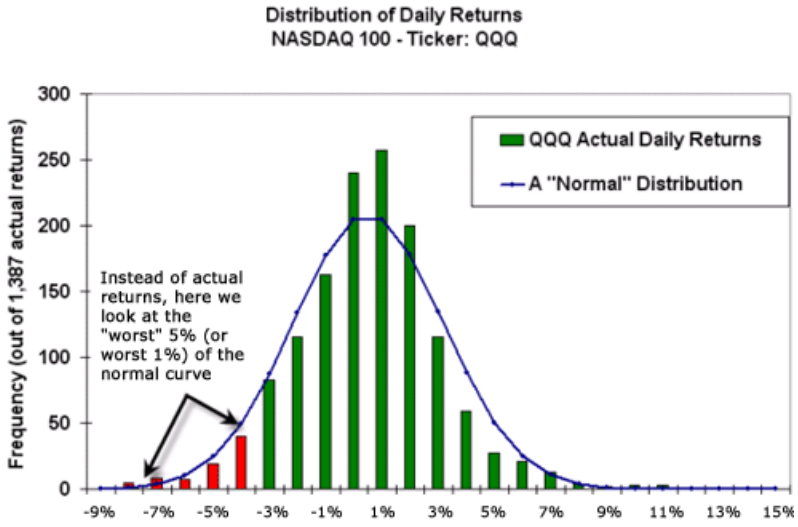


FIGURE 6-2 - VAR EXAMPLE

There are far too many portfolio performance measurement metrics to list, but the system will use simple equations (like the Sharpe Ratio [6]) to provide the user with useful information.

Stock market prediction involves very complex yet unreliable methods. This irregularity is due to the human factors at work in the stock market, such as fear and competition. Sometimes humans do not do what is best for them, mainly due to emotional influence. Nevertheless, one can try to predict stock prices using established mathematical models and methods.

One method that is in use today is artificial neural networks [7]. This involves having multiple computers approximate decision-making when given a large enough data sample. This kind of prediction is quite complex in itself and is beyond the scope of this project.

However, it is possible to use simpler models for price prediction. For example, one can use a linear or quadratic curve fit to a set or part of a set of historical data.

7) USER INTERFACE DESIGN

The goal of the user interface is to make the site look good and easy to use. This can be achieved using relatively simple web pages and forms to display and capture data, with a minimal amount of clicks/keystrokes. The typical user would point his/her browser to the game's website, log in, and start playing. No special equipment is needed, just a computer with an internet connection.

a) Preliminary Design

Sample Navigational Paths (all cases assume user is already on website's front page):

Start -> Log In -> View Portfolio -> Logout

Start -> Log In -> Manage Portfolio -> Buy Stock -> Logout

Start -> Log In -> Manage Portfolio -> Sell Stock -> Logout

Start -> Log In -> My Account -> Edit Notifications -> Configure -> Logout

Start -> Log In -> My Account -> Disable -> Logout

Start -> Log In -> Edit Advertisements -> Logout (Advertisers Only)

Start -> Log In -> My Account -> View history -> Logout

Start -> Log In -> View Recommendations -> Logout

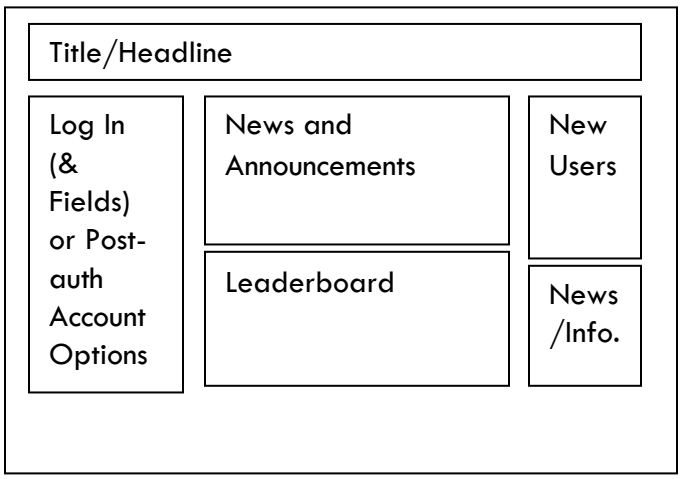
Start -> Log In -> View Recommendations -> Manage Portfolio -> Buy/Sell Stock -> Logout

Start -> View Leaderboard

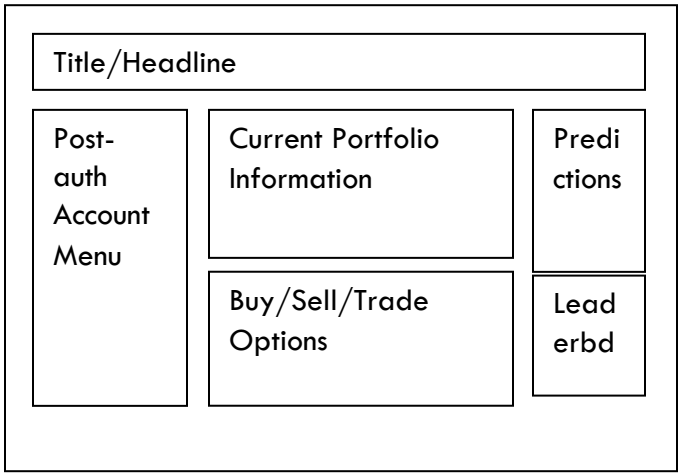
Start -> View New Users

...etc.

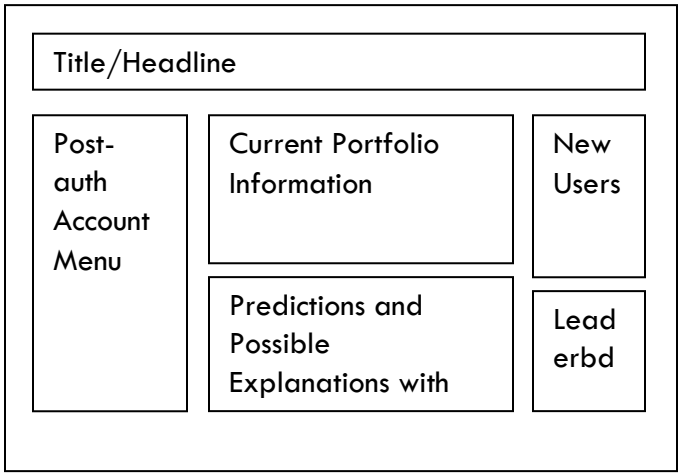
Sample Frontpage:



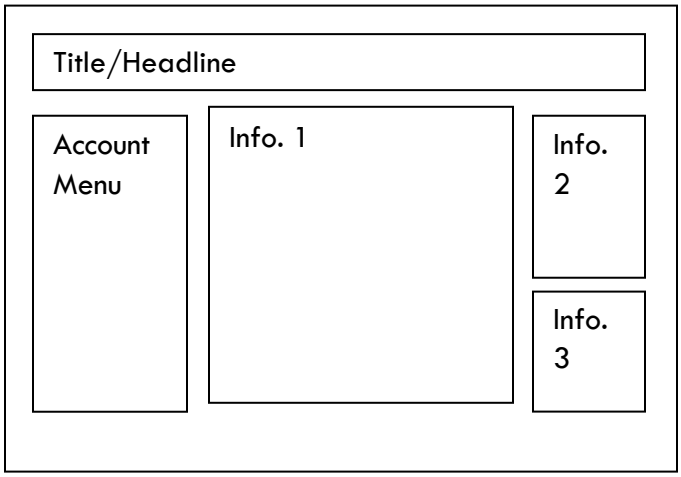
Sample Portfolio Page:



Sample Predictions Page:



Generic Template Page for all Other Pages:



Details on what each user needs to do and information he/she needs to enter can be found in the Effort Estimation section below. The general design is efficient and has been used on many popular sites.

b) User Effort Estimation

All of the following GOMS (Goals, Operators, Methods, and Selection Rules) Methods are done with the currently available knowledge of the site - they may change when the site is implemented. These methods are modeled loosely after the ones found on the page about GOMS [3].

NOTE: Keystrokes for data entry are omitted because they are variable. The minimal keystrokes/clicks for a given method ignoring data entry are described below. It is evident that most of the work the user has to do is data entry related, with minimal UI overhead.

Method for Goal: Log In/Authenticate - requires at least 2 clicks and 1 keystroke

1. Move cursor to Username text field and click mouse (context)
2. Enter username into text field using keyboard
3. Press "TAB" to move to password field
4. Enter password into text field using keyboard
5. Click "Log In" button
6. Return with goal accomplished

Method for Goal: Log Out of Account - 1 mouse click

1. Move cursor to "Log Out" link on page (context)
2. Click mouse button
3. Return with goal accomplished

Method for Goal: Create New Account - requires at least 3 clicks and 1 keystroke

1. Move cursor to "Create New Account" link on page and click mouse (context)
2. Click on the first input field
3. Enter desired information
4. Press "TAB" to go to next field
5. Repeat steps 4 and 5 until form is complete
6. Submit form by clicking "Submit" button
7. Return with goal accomplished

Method for Goal: Add to/Purchase stocks from market for a portfolio - requires at least 4 clicks and 1 keystroke

1. Accomplish goal: Log In/Authenticate
2. Move to and click on "Manage Portfolio"
3. Move to and click on "Buy Stocks"
4. Move to and click on stock ticker text field
5. Enter stock name/ticker symbol
6. Press "TAB" to move to next field
7. Repeat 5 and 6 for all required fields (likely ticker symbol, number of shares/$ amount, when to purchase)

8. Click "Submit"
9. Return with goal accomplished

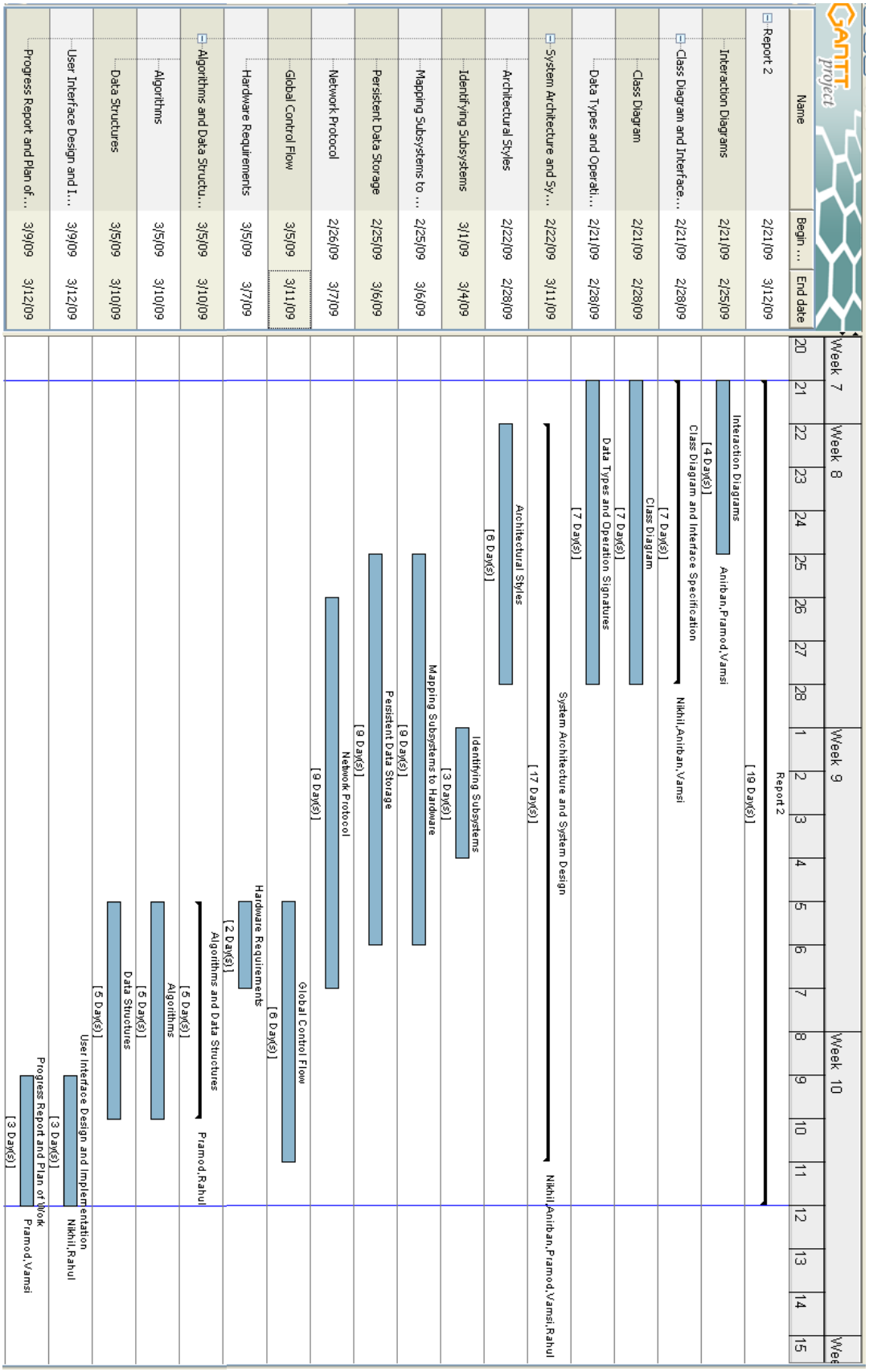
Method for Goal: Remove/Sell stock to market from a portfolio - requires at least 5 mouse clicks

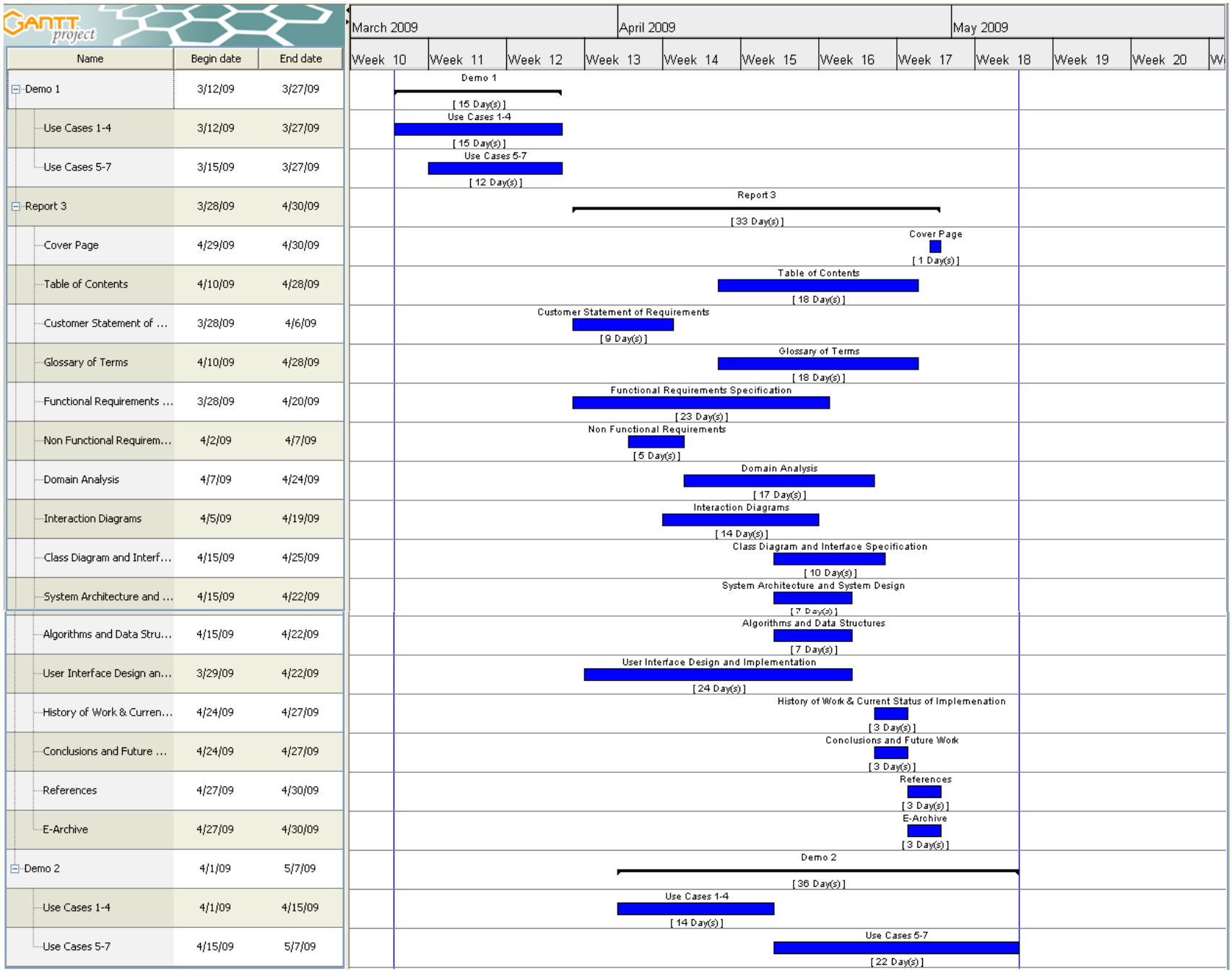
1. Accomplish goal: Log In/Authenticate
2. Move to and click on "Manage Portfolio"
3. Move to and click on "Sell Stocks"
4. Select desired stocks using checkboxes and mouse
5. Move to and click on "Confirm Sale"
6. Read page information, confirm as correct, then move and click "Submit" button
7. Return with goal accomplished

Method for Goal: Setup a New E-Mail/SMS Notification - requires at least 5 mouse clicks

1. Accomplish goal: Log In/Authenticate
2. Move to and click on "My Account"
3. Move to and click on "Edit"/"Manage Profile"
4. Move to and click on "Notifications"
5. Recall desired stock ticker symbol
6. Enter ticker symbol into text field
7. Move to and click on "E-Mail" and/or "SMS"
8. Move to and click on "Submit"
9. Return with goal accomplished

8) PLAN OF WORK




GANTT project
March 2009
April 2009
May 2009
Week 10
Week 11
Week 12
Week 13
Week 14
Week 15
Week 16
Week 17
Week 18
Week 19
Week 20
Name
Begin date
End date
Demo 1 3/12/09 3/27/09
Use Cases 1-4 3/12/09 3/27/09
Use Cases 5-7 3/15/09 3/27/09
Report 3 3/28/09 4/30/09
Cover Page 4/29/09 4/30/09
Table of Contents 4/10/09 4/28/09
Customer Statement of ... 3/28/09 4/6/09
Glossary of Terms 4/10/09 4/28/09
Functional Requirements ... 3/28/09 4/20/09
Non Functional Requirem... 4/2/09 4/7/09
Domain Analysis 4/7/09 4/24/09
Interaction Diagrams 4/5/09 4/19/09
Class Diagram and Interf... 4/15/09 4/25/09
System Architecture and ... 4/15/09 4/22/09
Algorithms and Data Stru... 4/15/09 4/22/09
User Interface Design an... 3/29/09 4/22/09
History of Work & Curren... 4/24/09 4/27/09
Conclusions and Future ... 4/24/09 4/27/09
References 4/27/09 4/30/09
E-Archive 4/27/09 4/30/09
Demo 2 4/1/09 5/7/09
Use Cases 1-4 4/1/09 4/15/09
Use Cases 5-7 4/15/09 5/7/09
Demo 1 [15 Day(s)]
Use Cases 1-4 [15 Day(s)]
Use Cases 5-7 [12 Day(s)]
Report 3 [33 Day(s)]
Cover Page [1 Day(s)]
Table of Contents [18 Day(s)]
Customer Statement of Requirements [9 Day(s)]
Glossary of Terms [18 Day(s)]
Functional Requirements Specification [23 Day(s)]
Non Functional Requirements [5 Day(s)]
Domain Analysis [17 Day(s)]
Interaction Diagrams [14 Day(s)]
Class Diagram and Interface Specification [10 Day(s)]
System Architecture and System Design [7 Day(s)]
Algorithms and Data Structures [7 Day(s)]
User Interface Design and Implementation [24 Day(s)]
History of Work & Current Status of Implemenation [3 Day(s)]
Conclusions and Future Work [3 Day(s)]
References [3 Day(s)]
E-Archive [3 Day(s)]
Demo 2 [36 Day(s)]
Use Cases 1-4 [14 Day(s)]
Use Cases 5-7 [22 Day(s)]

9) REFERENCES

Following is a list of references that have been used for any one of the parts of this report.

1. Existing Stock Market Game Websites: www.computer-game.us , www.shareme.com, www.filegets.com

2. FURPS

   "FURPS." Wikipedia, the free encyclopedia. 10 Feb. 2009 <http://en.wikipedia.org/wiki/Furps>.

3. GOMS

   "Tutorial GOMS (Kieras)." Home — College of Computing. 15 Feb. 2009 <http://www.cc.gatech.edu/computing/classes/cs6751_98_fall/handouts/GOMS-Kieras.html>.

4. UML 2.0 in a Nutshell

   Pilone, Dan, and Neil Pitman. UML 2. 0 in a Nutshell. Danbury: O'Reilly Media, Incorporated, 2005.

5. Introduction to Value at Risk (VaR)

   " Introduction to Value at Risk (VaR)." Investopedia. 19 Feb. 2009 <http://www.investopedia.com/articles/04/092904.asp>.

6. Sharpe Ratio

   "Sharpe Ratio." Investopedia. 19 Feb. 2009 <http://www.investopedia.com/terms/s/sharperatio.asp >.

7. Stock Market Prediction

   "Stock Market Prediction." Wikipedia, the free encyclopedia. 19 Feb. 2009 <http://en.wikipedia.org/wiki/Stock_market_prediction>.