The Paramount Investments League

Report 2
Software Engineering
14:332:452

Team 1:

David Patrzeba
Eric Jacob
Evan Arbeitman
Christopher Mancuso
David Karivalis
Jesse Ziegler

March 2, 2014

Hyperlinks:

Webapp Link
Project Repository
Reports Repository

Revision History:

Version No.	Date of Revision
v.2.1	3/2/2014

Contents

1 System Interaction Diagrams

1.1 Introduction

PLACEHOLDER TEXT –REPLACE WITH OUR OWN INTRO–

The following is an analysis of the interaction of UC-1 to UC-5. Within these use cases we touch on two of the most integral parts of our system, the persistant database, and the Yahoo! Finanace API adapter. The diagrams clearly describe the interactions of different subsystems to satisfy the use cases intent.

1.2 Diagrams

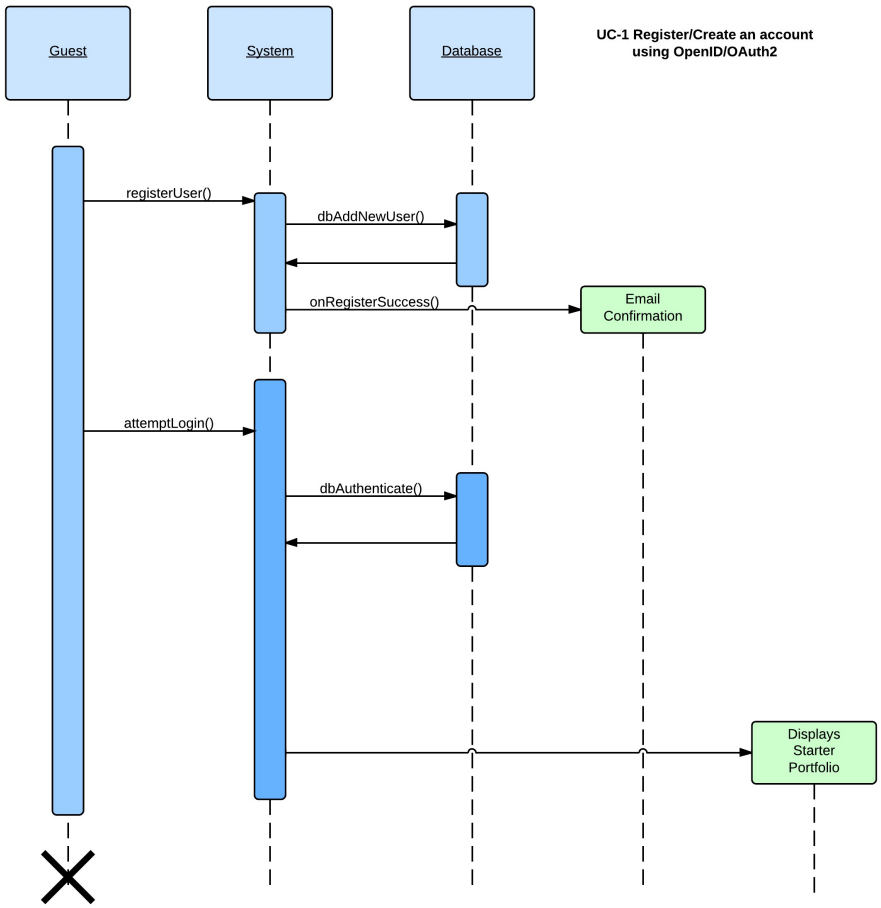

Figure 1.1: Shown in the sequence diagram for UC-1 begins with two options for the Guest. Either login or register an account. If a user attempts to register a new account the system is contacted with the users information. Then the system can attempt to check to make sure no duplicate login information exists in the database and if not it will store the new user information into the database. After this happens the user will be sent a confirmation email. If a user attempts to login, the system will attempt to authenticate the login details with details found in the database. If the details match correctly then the system will send the guest into investor mode and continue to upload their portfolio setting.

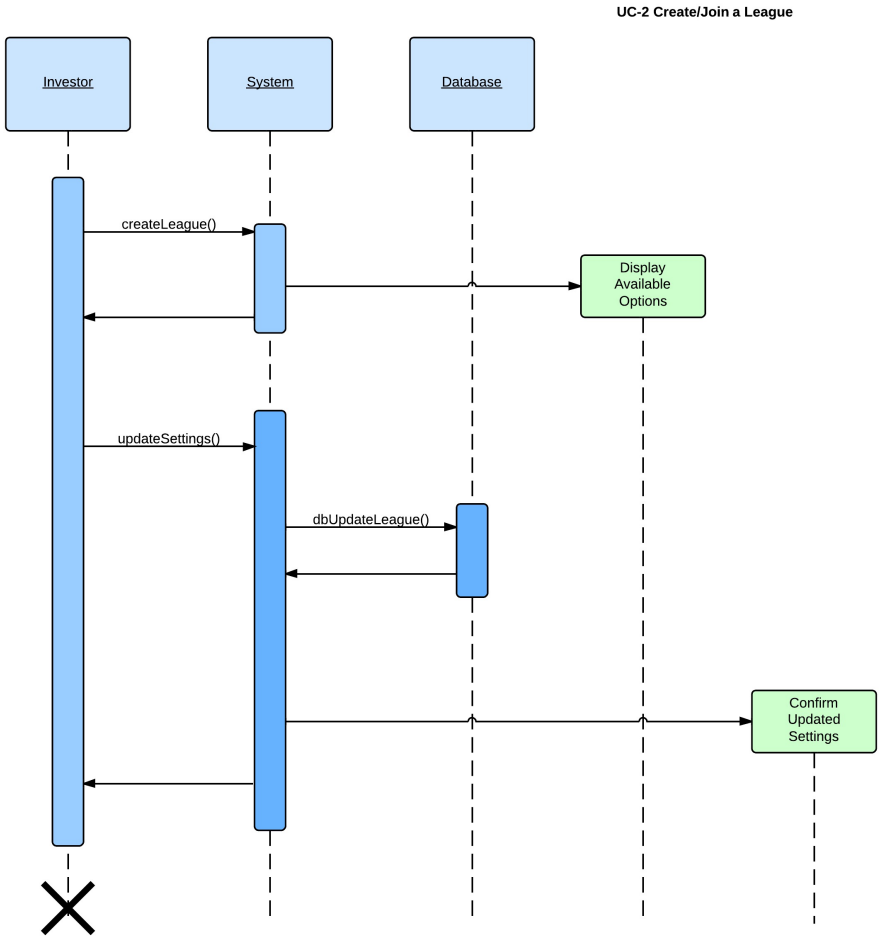

Figure 1.2: Shown in the sequence diagram for UC-2 is the flow of how to create an investment league. When an investor selects to create a league the system and more specifically the league controller will be contacted. The system will display the available options for creating a league. After there is a function updateSettings() which will create the league and process it in the database and also allow settings to be updated for a league. Not shown in the diagram is the alternative case of joining a league. The process to join a league is straightforward, where the league controller will show available leagues and then if an investor chooses to join they will be entered into the list in the database to associate with this league.

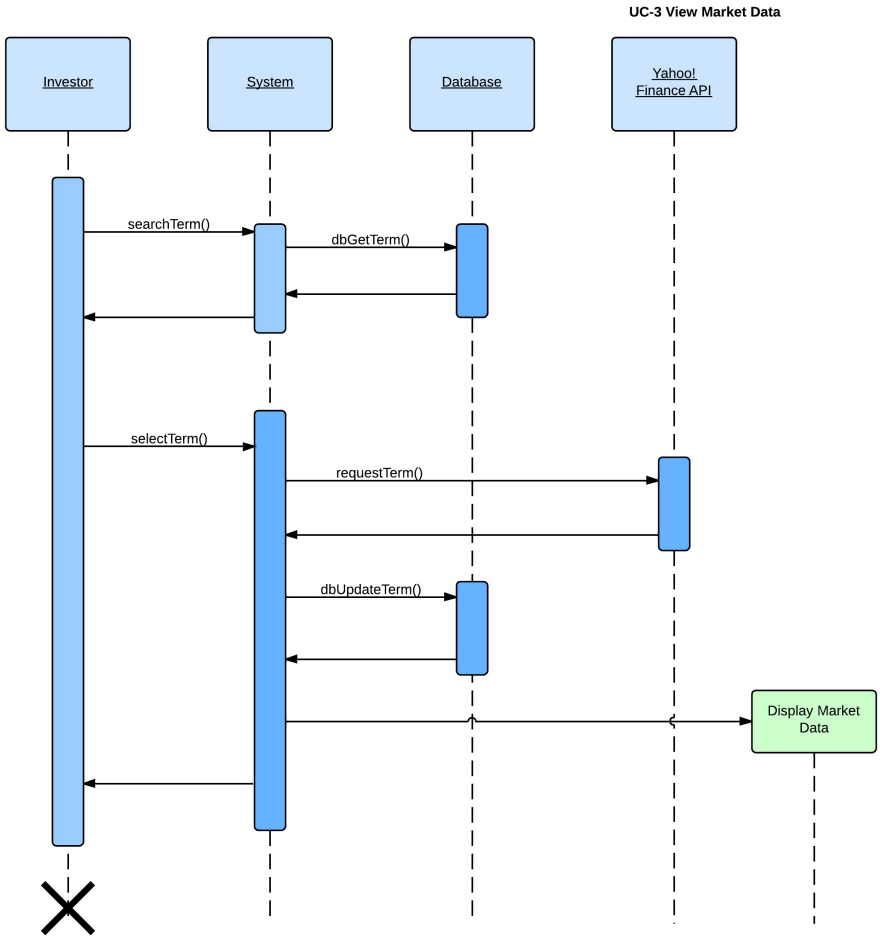

Figure 1.3: Viewing market data is accomplished by an investor searching a term. The system then finds this term which is most likely a company name or stock symbol. The system will fetch matches from the database and display them from the user. The investor will choose a match. The system takes the chosen term and requests its data from the Yahoo! Finance API. The system will update the database for this term, and then continue to display its market data to the investor.

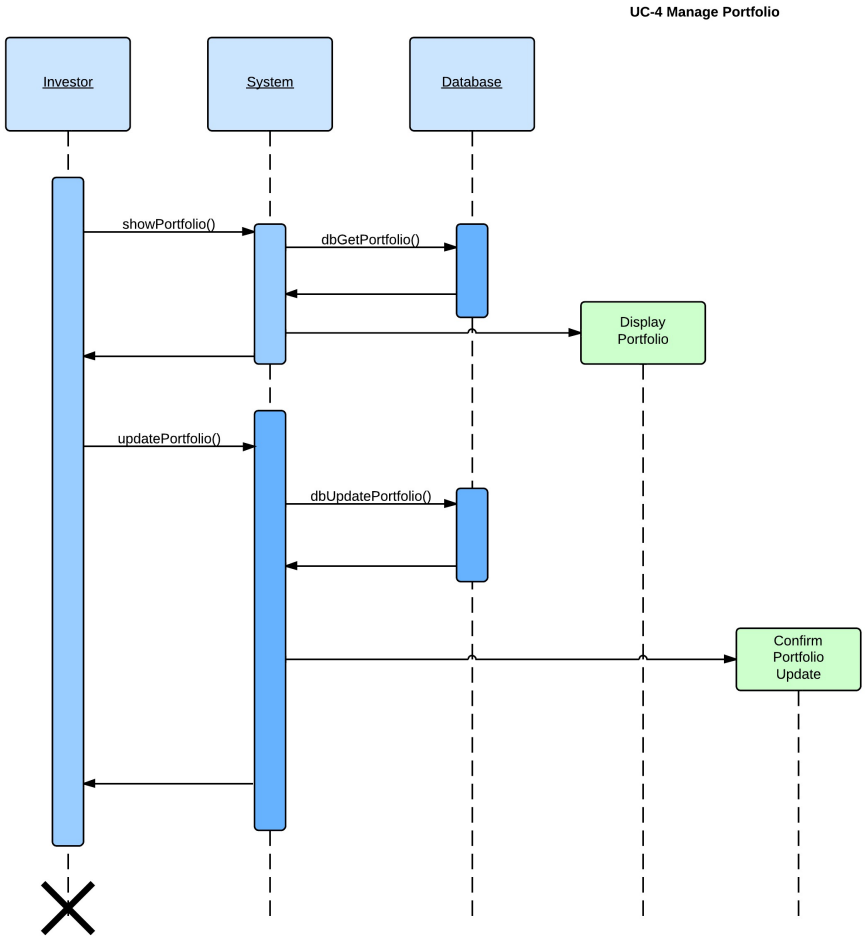

Figure 1.4: The investor should be able to view and make changes to their portfolio. The investor should be able to view their portfolio. The system will fetch the investors portfolio stocks from the database. The investor can also update their view of the portfolio and other settings.

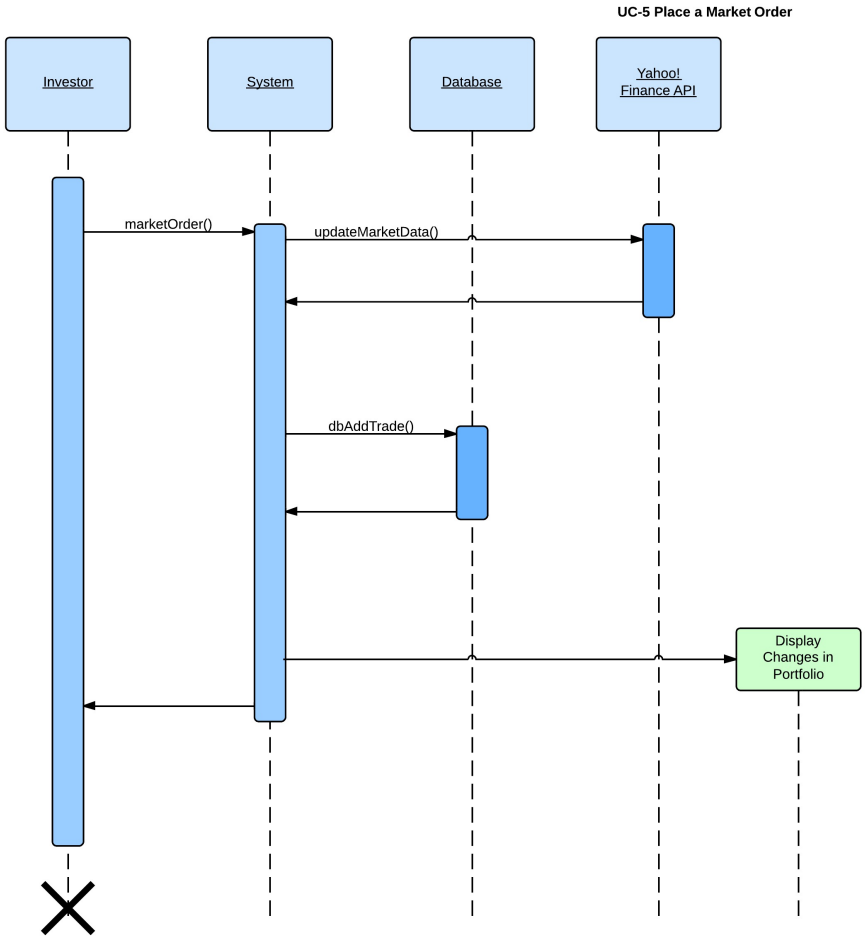

Figure 1.5: The investor needs to be able to place market orders. As soon as the investor places an order the system contacts Yahoo! Finance API to retrieve the current price of the stock. After the current price is found the system must confirm with the database that the user has enough funds to make a buy offer or enough stock to make the sell offer. After the trade is confirmed information will be stored about it in the database and the changes will be displayed in the investors portfolio.

2 Plan of Work

2.1 Development and Report Milestones

Illustrated on the next page is a chart reflecting our goals relative to the project dead-lines. It incorporates both core development and report items. For our initial stages we focus on environment and platform set-up (eg: deploying a development webserver) and the initial, core code implementation. At the same time we will finalize the details of our final product via the report milestones.

Development milestones have been spread out following the completion of Report 1 on 23 February 2014. It begins with deploying our development environment and server through Digital Ocean[1]. We concurrently will roll out developer images, the Play Framework[2], and develop database schema. Implementing user registration/login will follow shortly along with deploying a solution to use the Yahoo! Finance API. The development milestone finishes up with the implementation of user portfolios along with basic market operations and basic achievements.

Report milestones are also set concurrently. As we begin to initialize our development environment, we will also build on top of and expand on previous reports to expand upon and fully realize the details of *P*aramount Investment League.

Core goals leading up to Demo 1 include establishing all core functionality for *P*aramount Investment League. This includes the following:

- **Play Framework deployment :** This includes basic site navigation, user login/registration, and Twitter Bootstrap deployment.
- **Setting a foundation for the database:** Schema should be built to be extensible to support future enhancements.
- **Implement the Yahoo! Finance API**
- **A functional user interface:** The user interface should function across multiple platforms with a focus on experience and expectations.

2.2 Breakdown of Responsibilities Introduciton

Contributions leading up to the completion of this report are covered in the "Contributions" in Chapter 7. For the future division of labor, we all plan on subdividing aspects of both the next reports as well as the development of the *P*aramount Investment League Demo 1.

2.3 Breakdown of Responsibilities

Core server deployment will be the repsonsibility of David Patrzeba. Eric Jacob will be responsible for the database rollout. David Patrzeba will also be responsible for the core software rollout on the server including git, Play Framework, nginx, and other core libraries and software. David Karivalis will be repsonsible for integrating Twitter Bootstrap into Play Framework.

Routing will be headed by Eric Jacob and assisted by Chris Mancuso and Evan Arbeitman.

User Interface will be done by David Karivalis and Jesse Ziegler and they will integrate the REST API[3] to facilitate dynamic views.

The rest of the development workload will be divied up based around the Model, View, Controller design pattern. David Patrzeba and Eric Jacob will focus on the controllers, David Karivalis and Jesse Ziegler will focus on the Views, and Evan Arbeitman and Chris Mancuso will focus on models. David P., David K., and Eric will be made available for technical advising.

David Patrzeba will be responsible for formatting the report. David Karivalis will be responsible for digitization of paper diagramming for all reports. Report duties will be divied up based on percieved strengths of the team and availability.

Overall project success will be decided with how well the MVC[4] component teams communicate and work with each other, as *P*aramount Investment League will rely on the interactivity between the Model, Views, and Controller portions of the architecture.

2.4 Projected Milestones

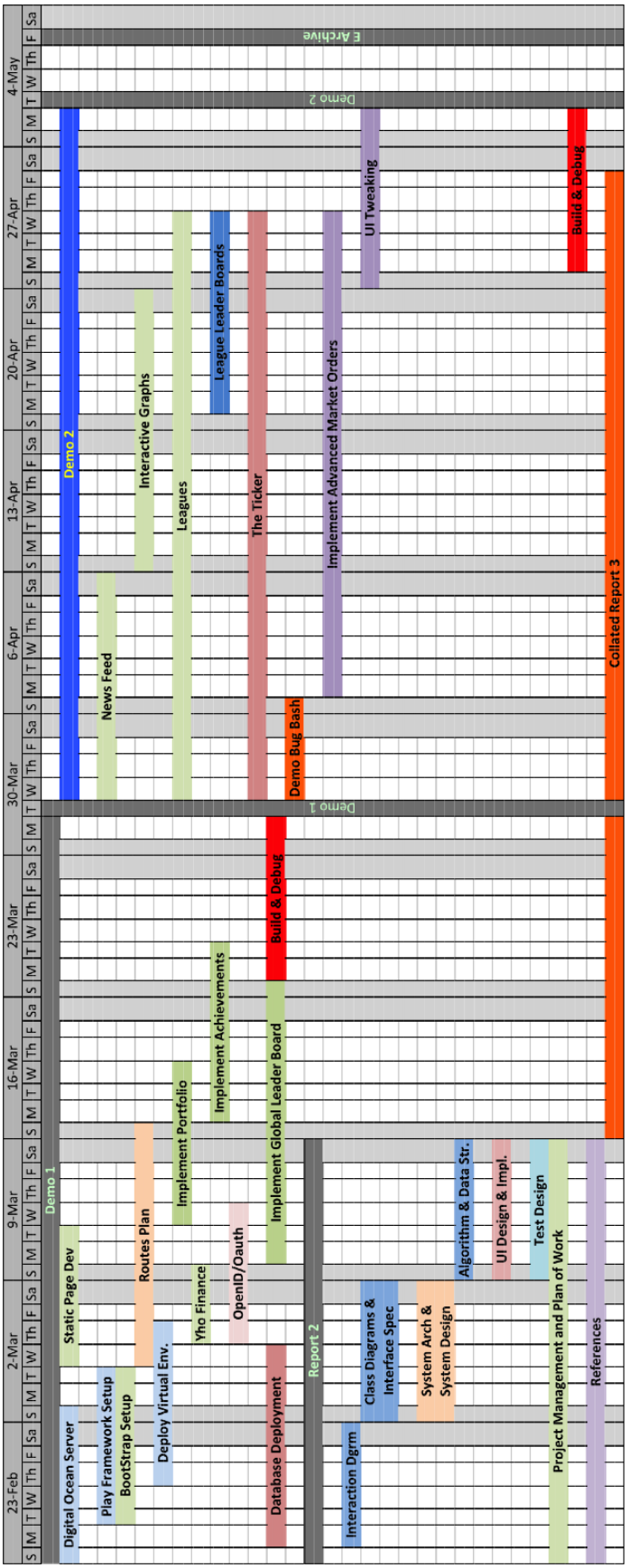

Figure 2.1: This chart is the roadmap to meeting all our milestones.

References

[1] Wikipedia, "Digital ocean." http://en.wikipedia.org/wiki/DigitalOcean. [Online; accessed 23 February 2014].

[2] Wikipedia, "Play framework." http://en.wikipedia.org/wiki/Play_Framework. [Online; accessed 23 February 2014].

[3] Wikipedia, "Representational state transfer - Wikipedia, the free encyclopedia." http://en.wikipedia.org/wiki/Representational_state_transfer. [Online; accessed 23 February 2013].

[4] Wikipedia, "Model-view-controller - Wikipedia, the free encyclopedia." http://en.wikipedia.org/wiki/Model-view-controller. [Online; accessed 23 February 2014].