

---

# **Software Requirements Specification**

**for**

**Bazhanga**

**Version 0.1**

**Prepared by Austin Brown, Jeffrey Caldwell, Tyler Duguay, Laura Kraus,  
John Maass, Jacob McIvor**

**Group 3**

**March 26, 2015**

# Table of Contents

<b>Table of Contents .....</b>	<b>ii</b>
<b>Revision History .....</b>	<b>ii</b>
<b>1. Introduction.....</b>	<b>1</b>
1.1 Purpose .....	3
1.2 Document Conventions .....	3
1.3 Intended Audience and Reading Suggestions.....	3
1.4 Product Scope .....	4
<b>2. Overall Description.....</b>	<b>5</b>
2.1 Product Perspective .....	5
2.2 Product Functions .....	5
2.3 User Classes and Characteristics .....	7
2.4 Operating Environment .....	8
2.5 Design and Implementation Constraints.....	9
2.6 User Documentation.....	9
2.7 Assumptions and Dependencies .....	10
<b>3. External Interface Requirements .....</b>	<b>12</b>
3.1 User Interfaces .....	12
3.2 Hardware Interfaces.....	14
3.3 Software Interfaces .....	28
3.4 Communications Interfaces .....	28
<b>4. System Features .....</b>	<b>29</b>
<b>5. Other Nonfunctional Requirements.....</b>	<b>30</b>
5.1 Performance Requirements.....	43
5.2 Safety Requirements.....	43
5.3 Security Requirements.....	43
5.4 Software Quality Attributes.....	43
5.5 Business Rules.....	44
<b>6. Other Requirements .....</b>	<b>45</b>
<b>Appendix A: Glossary.....</b>	<b>45</b>
<b>Appendix B: Analysis Models .....</b>	<b>47</b>
.....	47
<b>Appendix C: Milestones .....</b>	<b>50</b>

## Revision History

Name	Date	Reason For Changes	Version
Jeffrey Caldwell	3/20/15	Second Draft	0.2
John S Maass	4/7/2015	Third Draft	0.3

## **List of Figures:**

**Figure 1.1 Graphic displaying the Bazhanga logo.**

**Figure 3.1 Diagram displaying screen navigation.**

**Figure 3.2 Start screen layout.**

**Figure 3.3 Help screen layout.**

**Figure 3.4 Settings screen layout.**

**Figure 3.5 Profile screen layout.**

**Figure 3.6 New Profile screen layout.**

**Figure 3.7 Portfolio screen layout.**

**Figure 3.8 Buy Screen layout.**

**Figure 3.9 Sell screen layout.**

**Figure 3.10 Trends Screen layout.**

**Figure 4.1 Case diagram for a user interacting with Bazhanga.**

**Figure 4.2 Help use case.**

**Figure 4.3 Exit use case.**

**Figure 4.4 Select Profile case.**

**Figure 4.5 Create New Profile case.**

**Figure 4.6 Delete Profile case.**

**Figure 4.7 Advance Time case.**

**Figure 4.8 Stock Trends case.**

**Figure 4.9 Search Stocks case.**

**Figure 4.10 Buy Stocks layout.**

**Figure 4.11 Sell Stocks case.**

# 1. Introduction

## 1.1 Purpose

Bazhanga is an educational stock market simulation application which allows a user to manage a portfolio of historic stocks. The app is designed to run on iPhone-5 or higher and will be available to users through the Apple app store for a nominal fee. This application will require the user to have an internet connection in order to operate.

The basic operation of the app is based on the user's interaction with their portfolio. The portfolio that the user manages will have a date associated with it from the past which determines what stocks the user can buy or sell. The simulation occurs by advancing the time and allowing the user to make decisions based on the changes in the stock prices. As the simulation occurs their portfolio's value will decrease, increase or remain the same. Whether they have successfully or unsuccessfully managed their portfolio will be depend on whether the net value of portfolio is greater than when they started the simulation.

The application is designed to teach people about purchasing and managing stocks. It assumes that each individual will have little to no experience with the stock market. The target demographic for this app is people with disposable incomes who wish to get into the stock market but are unfamiliar with it. The users only input will be in the form of buying or selling stocks with the end goal of making a monetary gain in the process. From this activity the user should gain valuable insight into when to sell a stock to maximize their profit or minimize their loss. Since the application is using historic stock data, the user will also gain insight into how to make and manage long term investments.

The stock data used in the app will be obtained from Yahoo's historic stock database. It will be focused on stocks listed on the New York Stock Exchange. Stock data will be obtained by the application as the user demands it during the simulation and stored on the user's device. Obtaining the data as the user needs it rather than having them store all of the data on their phone at one time has the benefit of reducing the size of the application on download.

The application will be built in Apple's X-code software using the Swift programming language. The design of the application is broken into three different parts: backend, frontend and a Swift bridge. The backend main function is to retrieve any stock data requested by the user, storing the user portfolio and managing any stock data that is loaded onto the phone. The frontend will consist of the GUI which will allow the user to search for stocks to buy, sell their current stocks or advance the date. And finally, the Swift bridge will connect both the backend and frontend together.

## 1.2 Document Conventions

Any words or terminology that the reader is unfamiliar with can be found in the Glossary located in Appendix A.

## 1.3 Intended Audience and Reading Suggestions

The audience for this document includes all persons involved in the Bazhanga project. Appendix C of this document lists key milestones in its development.

Section 1 provides a brief introduction of the project and section 2 is a general description of most aspects of the project. These two sections should be read by some who wishes to obtain working knowledge of the project.

Section 3 describes the interface requirements for the application and section 4 describes the system requirements of the application. These section should be read by people who want to understand the features of the application in more detail.

Section 5 covers most nonfunctional requirements . These topics include performance requirements, safety requirements, security requirements, software quality attributes and business rules.

## **1.4 Product Scope**

Bazhanga is an educational stock market application designed to teach people about the difficulties of using the stock market successfully.

The set goals of this software are:

- To approximate the buying and selling of stocks on the stock market.
- To demonstrate some of the key concepts of the stock market.
- To show which strategies of buying and selling stocks have the best chance of increasing the user's wealth.
- To entertain while educating the user about the stock market.

## 2. Overall Description



**Figure 1.1** Graphic displaying the Bazhanga logo.

### 2.1 Product Perspective

Bazhanga is a stock market portfolio management simulator designed for Apple's iPhone. Although Bazhanga is dependent on an external database run by Yahoo for its stock data, the application itself is built entirely in Apple's Xcode IDE and has no direct reasonability for any external dependencies. With that being said, this document focuses on design and functionality of the application on the user's iPhone.

### 2.2 Product Functions

Below is listed the major functions that either the user or the application must perform. This section is meant to give a brief outline of said functions.

**2.2.1 User Profile**

- Keep user's identification information; i.e. name of profile
- Store users bank balance
- Current date of profile
- Stocks that are currently owned by the user
- Current price of all stocks owned by the user
- Past stocks owned by the user
- Store users net wealth of time

**2.2.2 Search for Stock**

- Allows user to search for stocks using stock symbol
- Inform user if the stock was not found
- View its historic stock price corresponding to the profile date

**2.2.3 Buy stock**

- Let user use money in bank account to buy stock
- If user does not have enough funds they are informed

**2.2.4 Sell Stock**

- The user can sell their stock at current historic market price
- Money is added to users bank account

**2.2.5 View Portfolio**

- Display list of all currently owned stocks
- Show current value and amount of all stocks

**2.2.6 View Bank Balance**

- Show current bank balance of user
- Display most recent transactions

**2.2.7 Advance Time**



- User can advance time to the next valid day
- Automatically updates all stock prices for stocks in user portfolio

### 2.2.8 Display Current Date

- Shows the current date for the users profile

### 2.2.9 Display Graph of Relevant Data

- Graph of a stock over time

## 2.3 User Classes and Characteristics

### 2.3.1 Classes

There are three classes of user.

The first class consists of people who have no experience with the stock market. They are picking up Bazhanga for the first time. They either have a personal curiosity about the stock market or an amount of real wealth that they feel they should invest.

The second class is made up of people who have experience with the stock market and no experience with Bazhanga.

The third class of users have experience with both and are using the app to find an advantageous strategy.

#### 2.3.1.1 Beginners who have no experience with the stock market.

2.3.1.1.1 This class of user is picking up Bazhanga for the first time. They are curious about the stock market and may or may not have money to spend on buying real stocks. They have not tried the stock market yet and may either have been scared off of the stock market by some source of information or are simply cautious and want a way to try it without risking real money.

2.3.1.1.2 They will likely experience the software as a game, using it to learn while also keeping themselves entertained.

#### 2.3.1.2 Beginners who have experience with the stock market.

2.3.1.2.1 This class of user has tried the stock market for real. They either lost money, broke even or did not see the growth they were hoping for.

2.3.1.2.2 Now, they are looking for an educational resource to learn from before their next attempt.

### 2.3.1.3 Advanced users who have experience with Bazhanga.

2.3.1.3.1 These users are people who have learned from Bazhanga and other sources and now wish to try out various strategies to see what might help them succeed.

2.3.1.3.2 They will read about different stock market strategies and utilize the software to see whether they work.

## 2.3.2 User Stories

### 2.3.2.1 Story 1

2.3.2.1.1 A user loads Bazhanga. They start a new profile. They are asked to enter a profile name. They enter a name in the text box, "Jim", and press the start button.

2.3.2.1.2 The main view loads. The user wants to buy Microsoft but doesn't know what its symbol is. The user types Micro in the search box and is presented with MSFT. They touch it to select it. This loads MSFT data to the main window.

2.3.2.1.3 The current price of MSFT is \$20.00. The user has \$100.00 as their bank balance. The user selects 10 shares and hits the buy key. The program creates a dialogue telling the user that they can't afford that many shares.

2.3.2.1.4 The user then changes the number of shares to 5. They hit the buy button. The user's bank balance becomes \$0.00 as they've bought 5 \* \$20.00 worth of shares. The user's portfolio updates to show 5 shares of MSFT.

2.3.2.1.5 The user advances time two months. Now the price of MSFT is \$40.00. The user decides now is a good time to sell their shares of MSFT. The user selects 5 shares and hits sell.

2.3.2.1.6 The user's portfolio is changed to show that they no longer have 5 shares of MSFT and their bank balance is now \$200.00.

## 2.4 Operating Environment

Bazhanga is designed to specifically run on iPhone-5 or higher and thus will only run in an iOS operating system. The application will be simple enough that an iPhone-5 can easily run it. The application will be built in the Swift 1.1 and may be susceptible to updates in the language.

This application will utilize Sqlite3 to store stock data and relevant user profile data. Sqlite3 will be accessed via the FMDB Objective-C wrapper. FMDB is open source and

free for download to anyone. Its license agreement stipulates that it is provided without any warranty and they are not responsible for any issue that may arise from its use.

## 2.5 Design and Implementation Constraints

The first design constraint stems from the fact that the amount of disk on some iPhones is only 16 GB. This severely limits the size of the application preventing us from storing all the stock data on the phone itself. To overcome this constraint we retrieve all of our stock data as it is needed from a database. This also means the user will always need an internet connection.

The size of every iPhone version limits how we can organize the user interface and display data in graphs or charts. If the interface is too cluttered the application may be rendered unusable. Alternatively, if it is too sparse it will appear unappealing to the user and they may abandon it before ever using it. We are going to have to find a middle ground such that the application's interface looks complex enough for someone to want to use it, but simple enough that it is easy to physically navigate.

## 2.6 User Documentation

In order to insure that all users of Bazhanga are adequately prepared to use the application a website with detailed tutorials on how to use the application will be maintained. The topics covered in the tutorials will be as follows:

### 2.6.1.1 Make a new profile

The user will be made aware of the limitations put on the number of profiles that exist at a time. It will also explain that each profile must have a unique name that can be edited by the user on creation. The options that the user has in choosing a starting date will also be explained. How to start with purchased stocks that are either completely random in nature or part of a particular scenario will be covered in this section.

### 2.6.1.2 Concept of Time and Advancing Time

This section will explain to the user that all stocks used in the application are historic, and thus the current time of any given profile will always be in the past and does not advance in real time. It will also explain how to advance time to the next trading day and that the user can choose to wait multiple days at a time.

### 2.6.1.3 Searching for and Finding Stock Prices

In this section, how to use the stock search feature to find stocks using a company's stock symbol will be explained. It will also show the user how to navigate the search results.

### 2.6.1.4 Buying Stocks

How choose how much of a certain stock the user wants to buy and how to buy shares of a stock will be explained. They will be made aware that they can only buy as many shares of stocks that they can afford.

#### **2.6.1.5 Selling Stocks**

Choosing how many shares of a given stock that the user wants to sell will be explained to the user. It will also be explained that all money made from the sale will be added to their bank account.

#### **2.6.1.6 Checking Bank Balance and Asset Value**

This section will explain to the user the importance of their bank balance and asset value. It will also explain how to navigate to them from any screen while play the simulation.

#### **2.6.1.7 Using Trends Graphs**

How graph the stock price for different stocks will be explained in this section along with how to interpret the graph will be explained to the user.

#### **2.6.1.8 Switching Profiles**

The user will be shown how to switch profiles in this section. It will also explain that all data and setting for any profile is saved automatically when it is closed.

#### **2.6.1.9 Deleting Profiles**

This section will explain how to delete a profile to the user. They will be made aware that any profile that is deleted cannot be restored.

#### **2.6.1.10 Closing Application**

The user will be shown how to exit the application from any screen that they may be on.

## **2.7 Assumptions and Dependencies**

### **2.7.1 Yahoo Finance! API**

All stock data used in the application will be retrieved from Yahoo Finance API as the user advances time and searches for stocks. Since Yahoo Finance is not in our control we cannot always guarantee that it will always be functional.

### **2.7.2 FMDB**

Since FMDB is an open source piece of software that is not maintained by us. We will be completely dependent on third parties to ensure that any bugs found while using it are fixed or any required updates needed to keep using it with future versions of Swift or iOS are made.

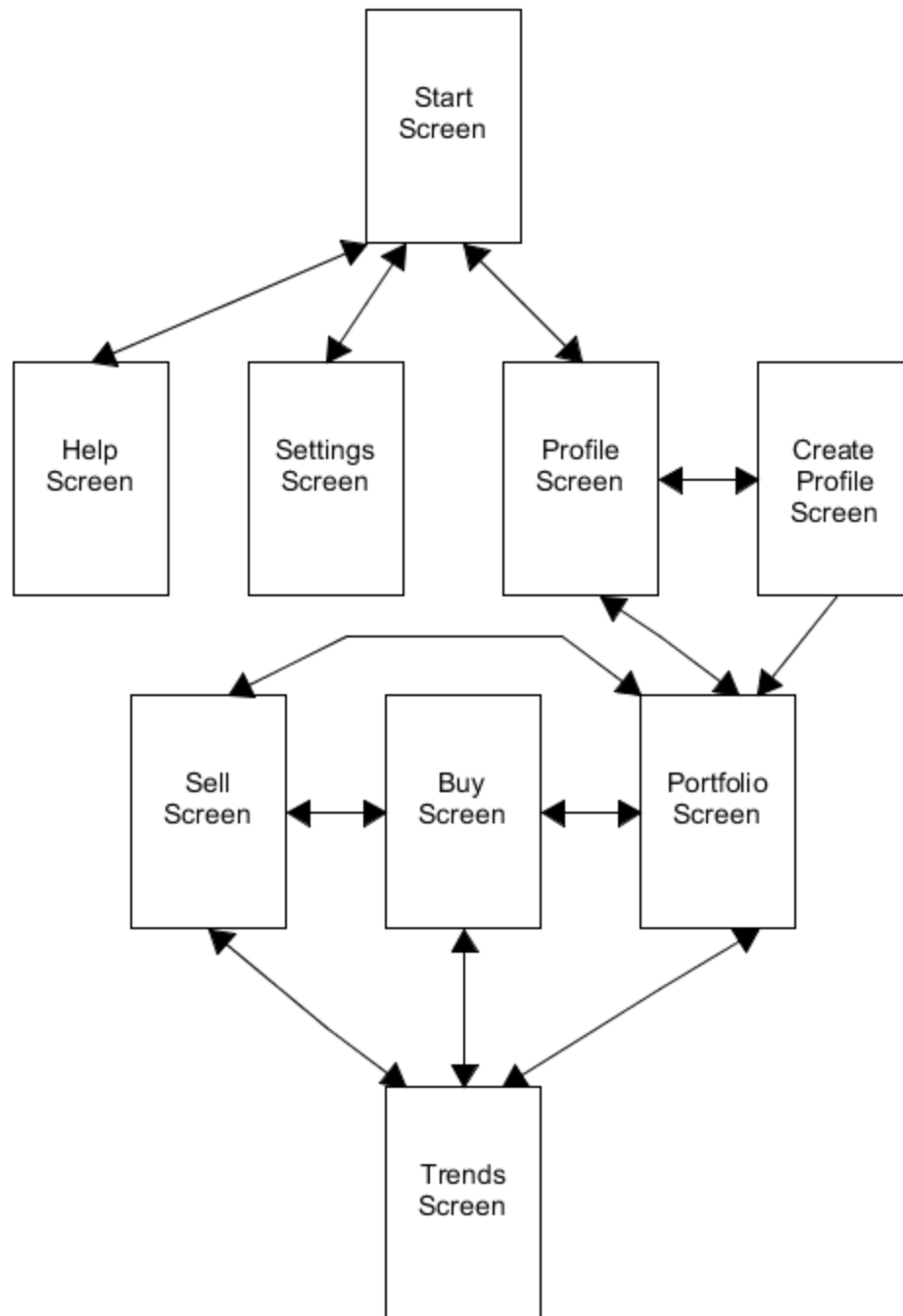
### **2.7.3 Internet Connection**

This application will always require an internet connection in order to retrieve new stock data from Yahoo Finance. There is no way around this dependence due to the overall size of all of the stock data required to for this application. Storing this data on the phone would be unreasonable and would most likely deter people from ever considering to download the application in the first place.

## **3. External Interface Requirements**

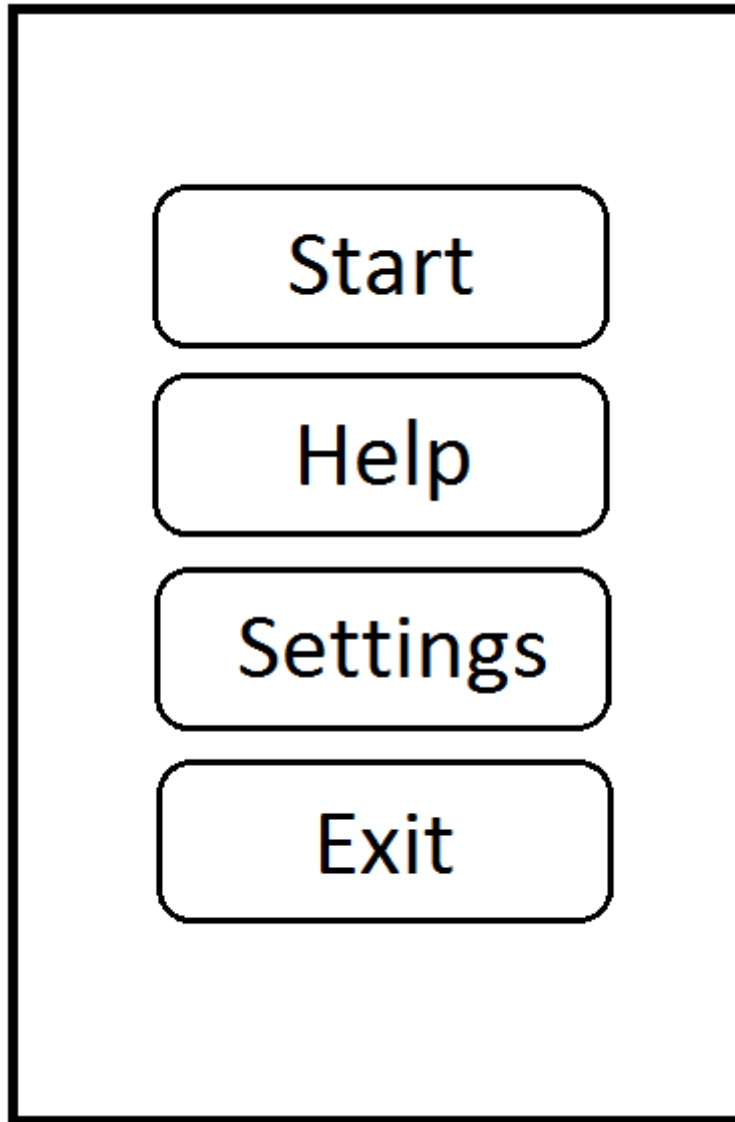
### **3.1 User Interfaces**

There are a total of nine different screens in the application that the user can access. A diagram showing how the user should be able to navigate through the different screens is shown in **Fig. 3.1**. Of the nine screens, only the Sell, Buy, Portfolio and Trends Screens are used in the simulation by the user. The remaining screens function to allow the user to manage their profiles, seek help for playing the simulation or change the color settings of the application.



**Fig 3.1** Diagram displaying how the user can navigate from one screen to another in the application.

### 3.1.1 Start Screen



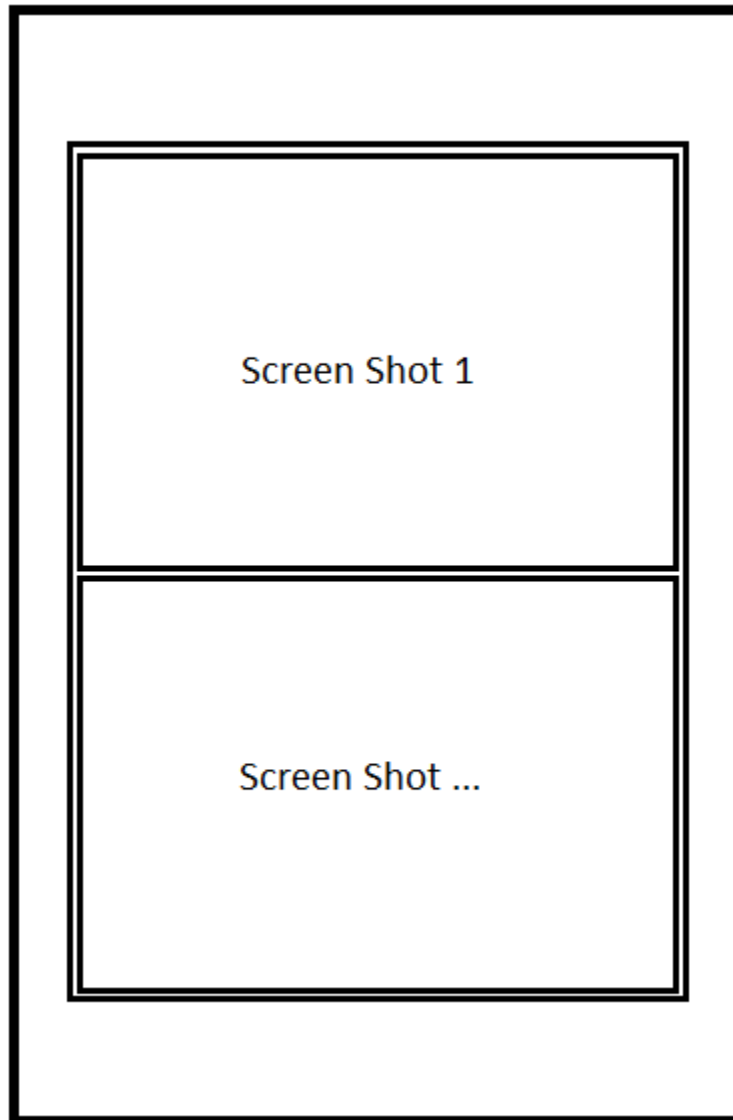
**Fig 3.2** Start screen layout.

**Key Features:**

- The Start Screen will load every time the user starts Bazhanga.
- From the start screen the user will be able to navigate the Profile, Help or Setting Screens as well as exit from the application entirely.
- The back button built into the phone will not have any function on this screen.
- The start screen can be reached from the Profile, Help or Setting Screens by hitting the back button built into the phone.



### 3.1.2 Help Screen

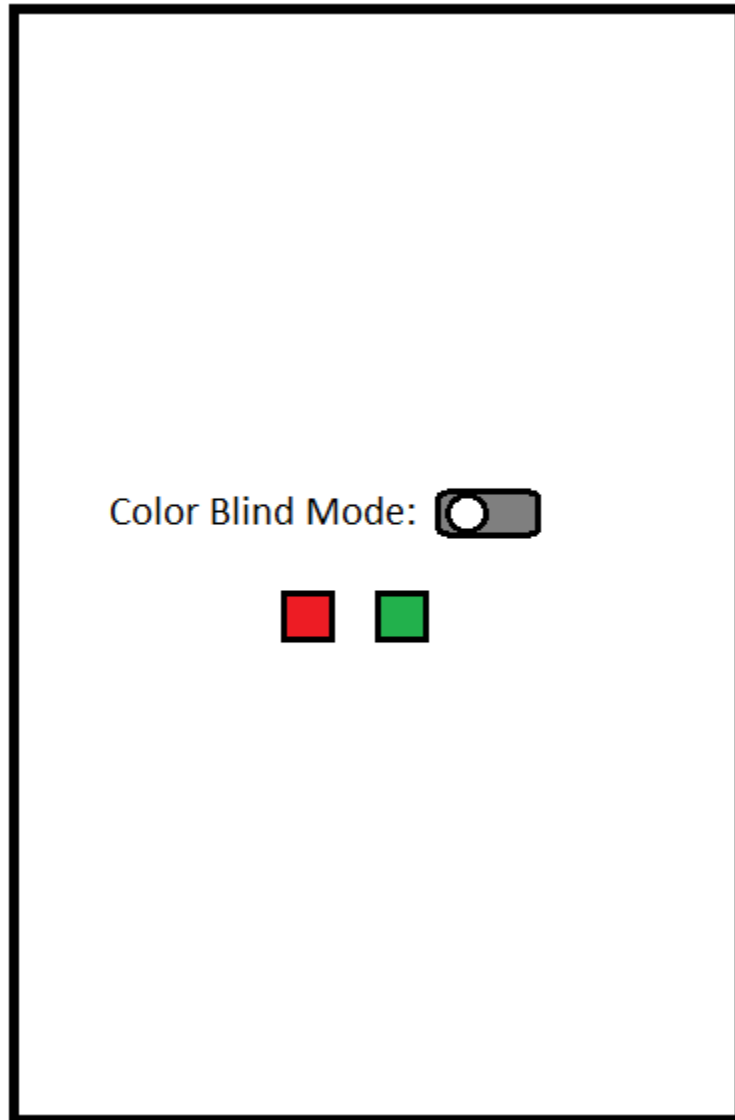


**Fig 3.3** Help screen layout.

**Key Features;**

- Once at the start screen, the user can only navigate back to the start screen using the built in back button on the phone.
- The help screen itself will contain a simple walk through that is comprised of screenshots and text to help the user with the most basic concepts of using Bazhanga.

### 3.1.3 Settings Screen

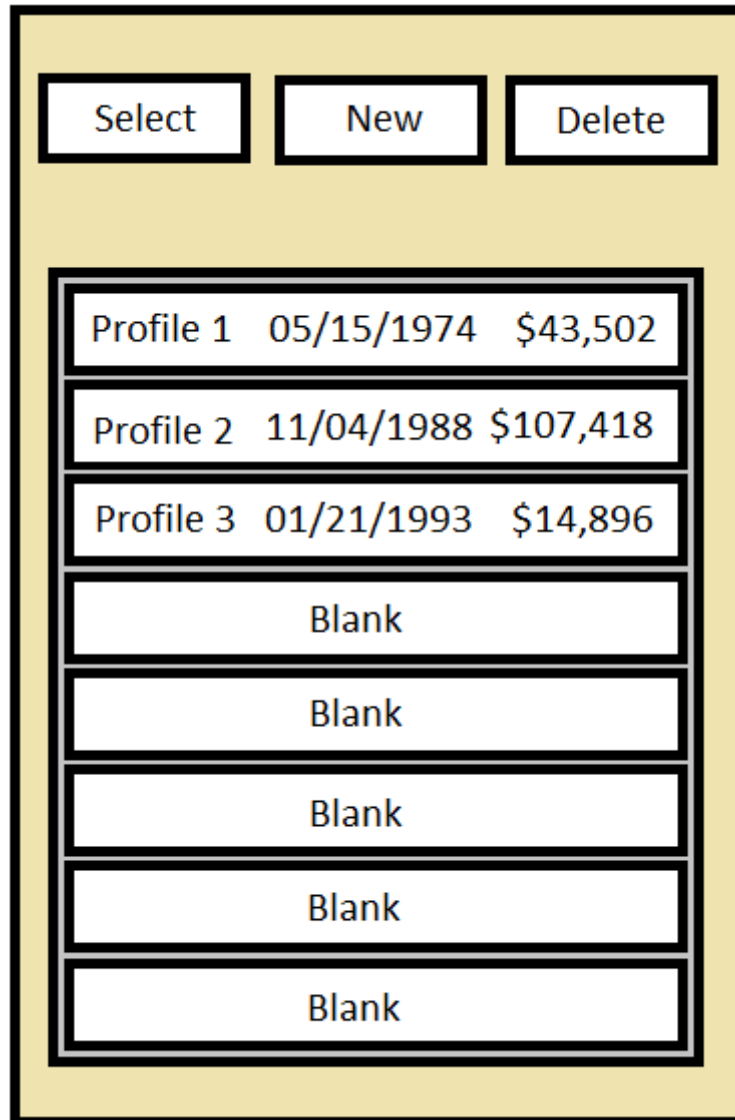


**Fig. 3.4** Settings screen layout.

**Key Features:**

- The user can set the color layout to color blind mode.
- Hitting the back button built into the phone will return the user to the start screen.

### 3.1.4 Profile Screen



The diagram illustrates the layout of the Profile screen. It features a yellow background with a black border. At the top, there are three buttons: 'Select', 'New', and 'Delete'. Below these buttons is a table with three columns: Profile ID, Date, and Amount. The table contains three rows of data and five rows of blank space for additional entries.

Profile ID	Date	Amount
Profile 1	05/15/1974	\$43,502
Profile 2	11/04/1988	\$107,418
Profile 3	01/21/1993	\$14,896
Blank		
Blank		
Blank		
Blank		
Blank		

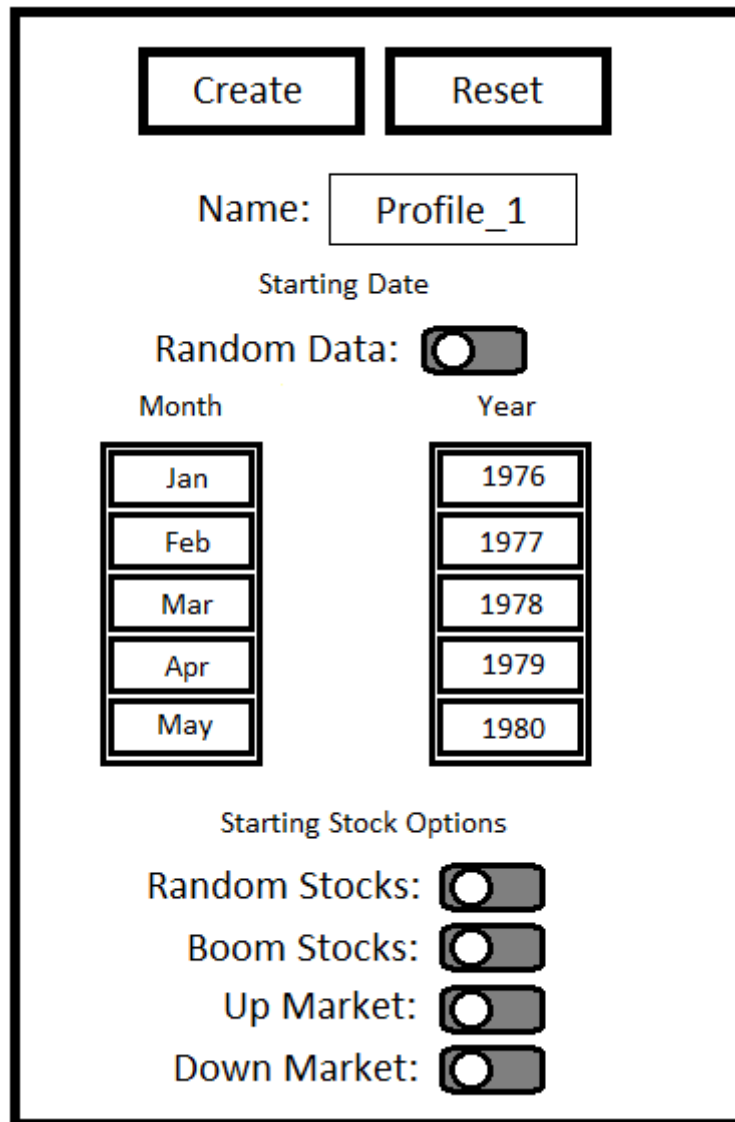
**Fig. 3.5** Profile screen layout.

**Key Features:**

- The user can navigate to the screen to create a new profile from here by hitting the new button.
- The user can delete an old profile by selecting it in the profile list and hitting the delete button.
- The user can continue playing a profile by selecting profile from list and hitting select button.

- This screen contains a list that shows the name, current date and total value of stocks that each profile has.
- Hitting the back button built into the phone will return the user to the start screen.

### 3.1.5 New Profile Screen



The diagram illustrates the layout of the 'New Profile' screen. At the top, there are two buttons: 'Create' and 'Reset'. Below these is a 'Name:' label followed by a text input field containing 'Profile\_1'. Underneath is the 'Starting Date' section, which includes a 'Random Data:' label and a toggle switch that is currently turned off. Below the toggle are two columns of selection buttons. The left column, labeled 'Month', contains buttons for 'Jan', 'Feb', 'Mar', 'Apr', and 'May'. The right column, labeled 'Year', contains buttons for '1976', '1977', '1978', '1979', and '1980'. At the bottom, the 'Starting Stock Options' section contains four labels with corresponding toggle switches: 'Random Stocks', 'Boom Stocks', 'Up Market', and 'Down Market'. All four toggle switches are currently turned off.

**Fig. 3.6** New Profile screen layout.

**Key Features:**

- The user can enter a unique profile name or use the default name.
- The user can pick a random starting date by hitting a switch or they can pick the month and year they wish to start in. The default date will always be the first valid date for the stocks stored in Yahoo's database if they choose to make no choice.
- If the user picked a random date they can also choose to start with either random stocks, Bubble stocks, stocks from an Up Market or stocks from a Down Market.

- If the user has picked a specific date or made no choice as to the date they wish to start then they can only pick to start with random stocks.
- The user can create the new profile only if they haven't exceeded the maximum number of profiles and the profile name is unique.
- Hitting the back button built into the phone will return the user to the start screen.

### 3.1.6 Portfolio Screen

The diagram illustrates the layout of the Portfolio screen. At the top, there are three buttons: "Portfolio", "Buy", and "Sell". Below these are three summary boxes labeled "Bank Account", "Asset Value", and "Networth", each displaying a placeholder value "\$XXXXX.XX".

Below the summary boxes is a table of stock holdings with the following columns: "Symbol", "Price", and "Owned". The table contains five rows of data:

Symbol	Price	Owned
GOOG	54.21	13
API	76.94	7
APP	33.01	32
DTE	89.18	11
GE	109.42	44

Each row in the table has a "Trends" button to its right. Below the table, the "Current Date" is displayed as "04/15/2003". A slider control is shown with a circle on the left and a diamond on the right, with the text "3 Days" next to the diamond. At the bottom, there are three buttons: "Inc", "Dec", and "Advance Time".

**Fig. 3.7** Portfolio screen layout.

**Key Features:**

- The Portfolio button at the top of the screen will be highlight indicating that the user is on the Portfolio screen.
- This screen displays the current value of the profiles bank account, asset value and net worth.
- All stocks that are currently owned by the user are displayed in a list.

- The user can choose the amount of time to advance by moving a slider to pick the number of days. Fine adjustments to the number of days to wait are made using two buttons which increment or decrement the slider by one. The default value will be one day.
- The user can advance time by hitting the Advance Time button.
- The user can also navigate to the Trends screen by hitting the Trends button for each stock in the stock list.
- Hitting the back button built into the phone will return the user to the profile screen.



### 3.1.7 Buy Screen

Portfolio Buy Sell

Current Date: 04/15/2003

G Search

Symbol	Price	Owned	
GOOG	54.21	14	Trends
GOQ	76.94	0	Trends
GR	33.01	0	Trends
GT	89.18	0	Trends
GXT	109.42	0	Trends

Bank Account: \$XXXXX.XX

3 Shares

Inc Dec Buy Shares

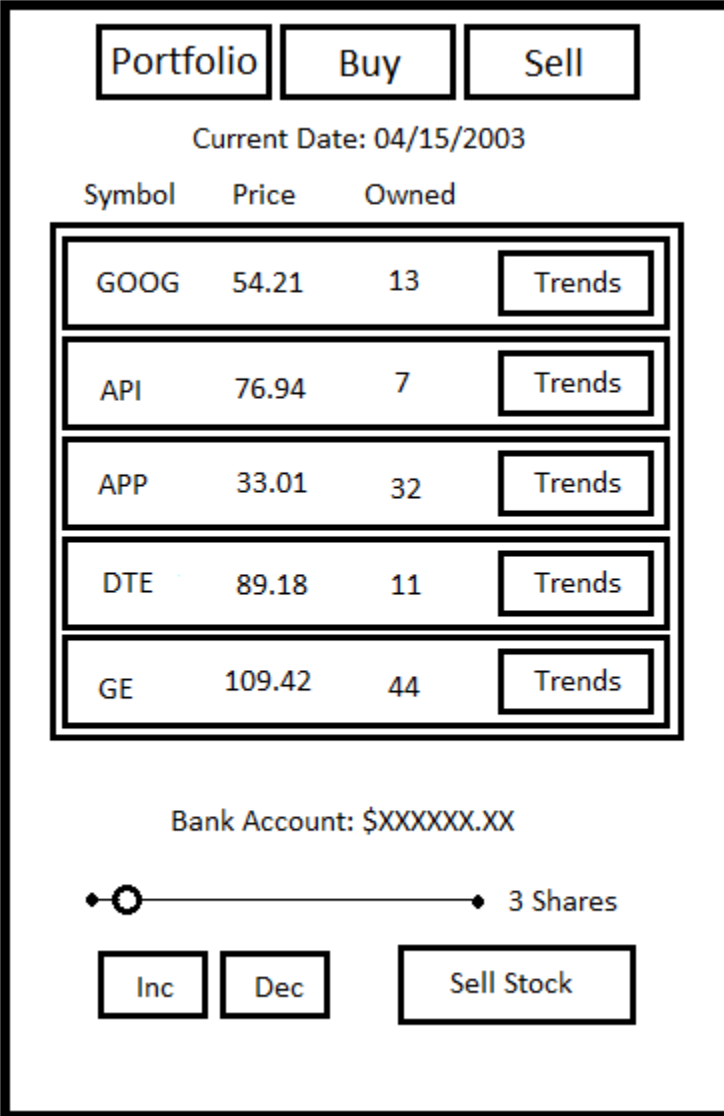
**Fig. 3.8** Buy Screen layout.

**Key Features:**

- The Buy button at the top of the screen will be highlighted indicating that the user is on the Buy screen.
- The user can search for a stock based on its stock symbol.
- The stocks that the user can purchase will be shown in a list that will only have twenty five stocks at a time.
- Hitting the next button will load the next stocks.
- Hitting the previous button will load the previous twenty five stocks.
- The price and amount currently owned of each stock will be listed.

- The user can select how many stocks they want to buy using a slider and two buttons to increment or decrement the slider by one. The maximum number of stock they are allowed to buy will always be how much they can afford for the current selected stock.
- By selecting a stock in the list, choosing how many to buy and hitting the Buy Shares button the user will purchase the selected stocks.
- The user can navigate to the Trends screen for each stock by hitting the trends button in the stock list.
- Hitting the back button built into the phone will return the user to the Profile screen.

### 3.1.8 Sell Screen



The diagram illustrates the layout of the 'Sell' screen. At the top, there are three buttons: 'Portfolio', 'Buy', and 'Sell'. The 'Sell' button is highlighted with a thicker border. Below these buttons, the text 'Current Date: 04/15/2003' is displayed. A table follows, with columns labeled 'Symbol', 'Price', and 'Owned'. The table contains five rows of stock data. Each row has a 'Trends' button to its right. Below the table, the text 'Bank Account: \$XXXXXX.XX' is shown. A slider control is positioned below the bank account text, with a circular knob on the left and a line extending to the right, ending at a point labeled '3 Shares'. Below the slider are three buttons: 'Inc', 'Dec', and 'Sell Stock'.

Symbol	Price	Owned	
GOOG	54.21	13	Trends
API	76.94	7	Trends
APP	33.01	32	Trends
DTE	89.18	11	Trends
GE	109.42	44	Trends

Bank Account: \$XXXXXX.XX

3 Shares

Inc Dec Sell Stock

**Fig. 3.9** Sell screen layout.

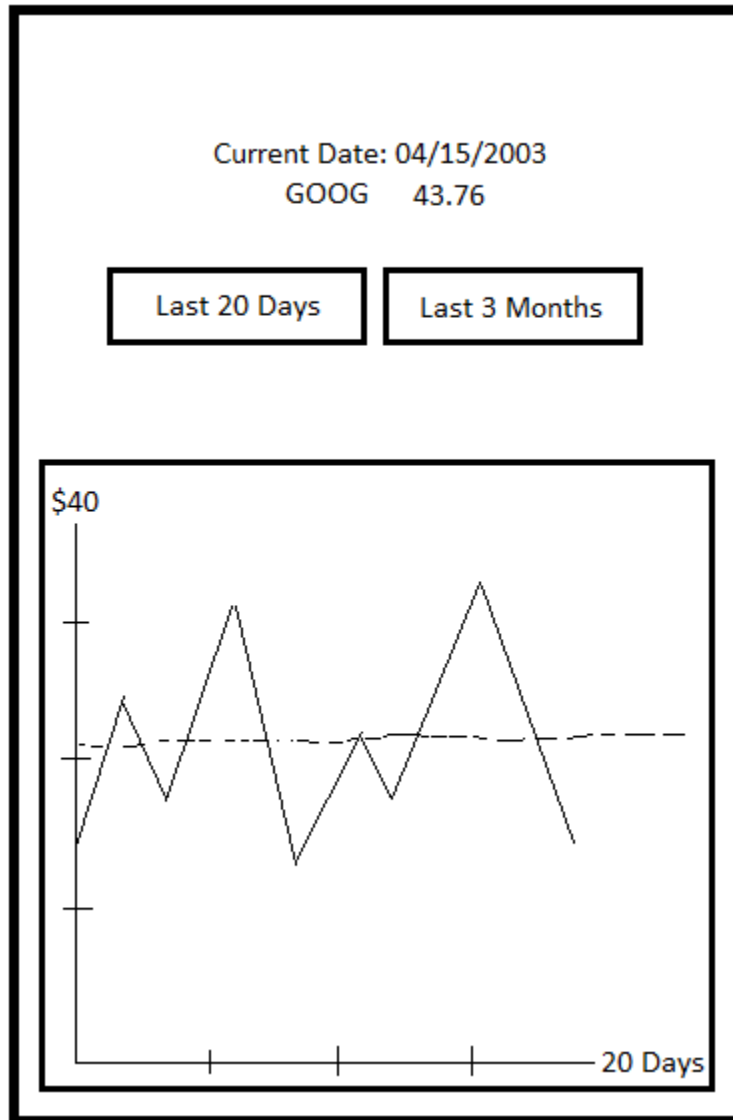
**Key Features:**

- The Sell button at the top of the screen will be highlighted indicating that the user is on the Sell screen.
- The screen contains a list that shows all of the stock currently owned by the user.
- Each list items shows the stock symbol, current price and amount owned.
- The user can navigate to the Trends screen for each stock.
- The user can select the amount of stocks they wish to sell using a slider and two buttons to increment or decrement the sliders by one. The maximum

number of stocks that a user can sell is limited the total amount of the currently selected stock.

- If the user chooses to sell a stock, the profit made from selling the stock is added to the users bank account.
- Hitting the back button built into the phone will return the user to the Profile screen.

### 3.1.9 Trends Screen



**Fig. 3.10** Trends Screen layout.

**Key Features:**

- The users can choose to display a graph showing the closing trading price for the last twenty days or three months of a stock.
- If the user owns the stock then a break even line is displayed on the graph. This line shows the minimum value that a stock needs to be in order for the user make a profit.
- Hitting the back button built into the phone will return the using the screen that they came from.

### **3.2 Hardware Interfaces**

Interaction between the user and the software is through the touch screen of the iPhone. There are no other hardware requirements for this application.

### **3.3 Software Interfaces**

The application will use the iOS operating system, Sqlite3 and other software that is found on the iPhone. Other than the FMDB wrapper that is used to communicate with Sqlite3, no software components that are not found natively on the iPhone will be used.

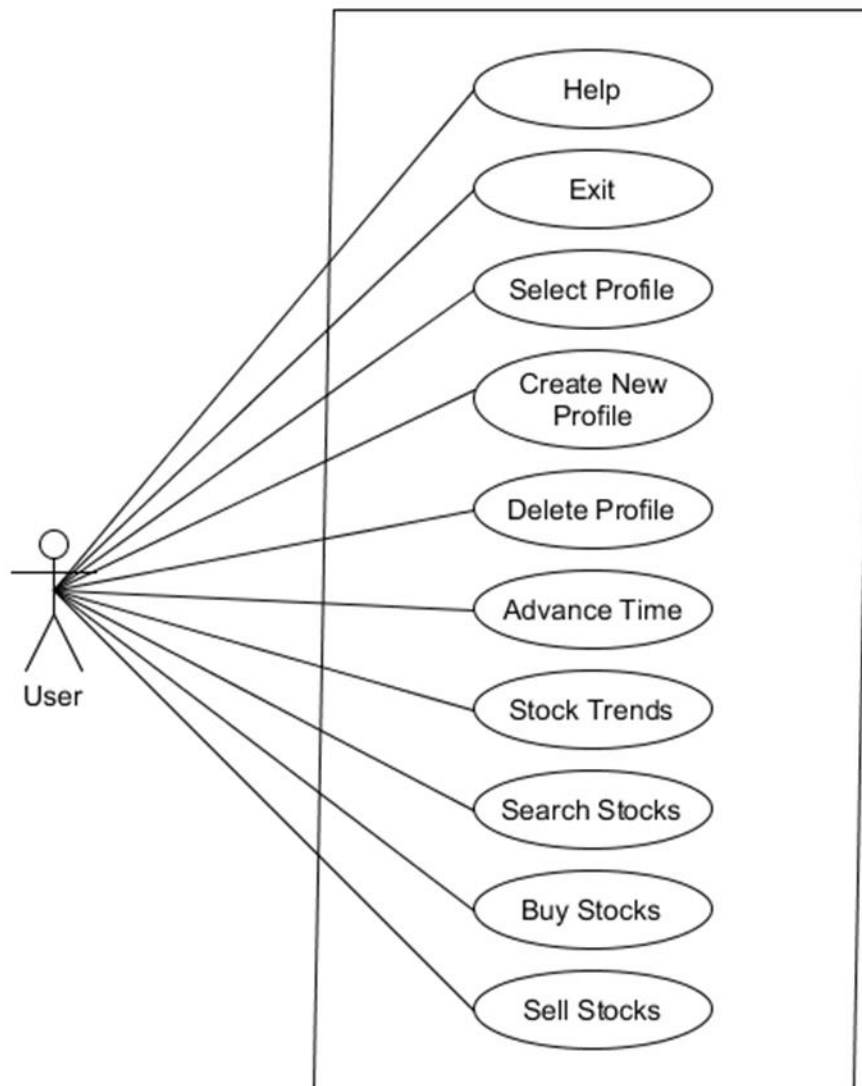
### **3.4 Communications Interfaces**

Communication between Sqlite3 and the application is accomplished through FMDB. How FMDB communicates with Sqlite3 and the application is not important for the creation of this application.

The application also has to communicate with Yahoo Finance to obtain stock data to run the simulation. This will require an internet connection in order to pull data from the database.

## 4. System Features

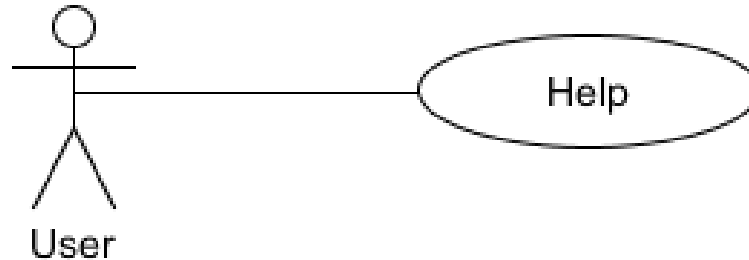
The System Features of Bazhanga are split into four different categories. The first category is based on the use case diagram shown in **Fig. 4.1**. The user interactions displayed in the diagram represent much of the core features of the application. The second category is derived from the requirements in order to maintain a profile and the third is defined by the implementation of a Sqlite3 database. The fourth category is made up of other features that are required for the application to run correctly, but do not fall into a specific category.



**Fig. 4.1** Case diagram for a user interacting with Bazhanga.

## 4.1 Use Case Functional Requirements

### 4.1.1 User Case: Help



**Fig. 4.2** Help use case.

#### 4.1.1.1 Description and Priority

Priority: Low

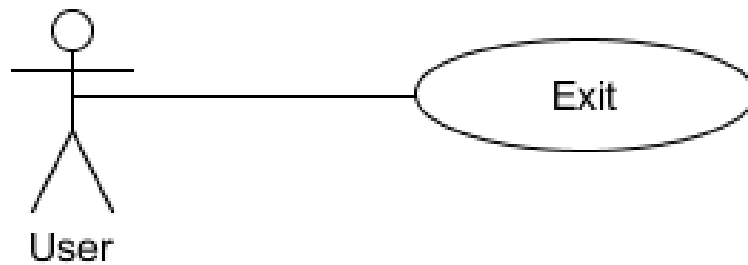
Description: The Help Screen contains a simple step by step walk through of basic functions such as setting up a profile, buy and selling stock, searching for stock and advancing time. The walk through will comprised of a series of screen shots and text in a list that the user can scroll through. The topics covered in the walk through will be ordered alphabetically.

#### 4.1.1.2 Response Sequence

- 1) User navigates to the start screen of the application.
- 2) User pushes the Help button.
- 3) User scrolls through walk through reading what information they find pertinent.



#### 4.1.2 User Case: Exit



**Fig. 4.3** Exit use case.

##### 4.1.2.1 Description and Priority

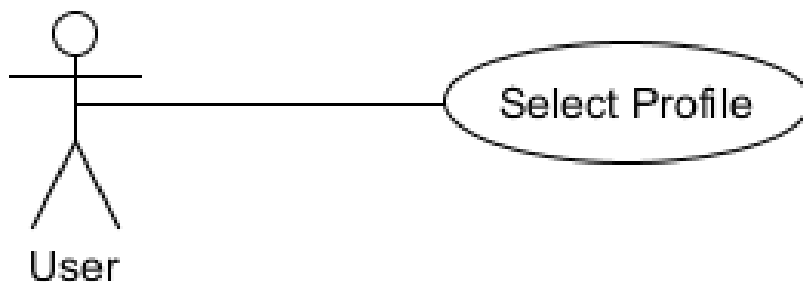
Priority: High

Description: The Exit button will completely close the application and any dependencies that it may be running.

##### 4.1.2.2 Response Sequence

- 1) User navigates to the start screen of the application.
- 2) User Pushes the Exit button.

#### 4.1.3 User Case: Select Profile



**Fig. 4.4** Select Profile case.

#### 4.1.3.1 Description and Priority

Priority: High

Description: The user selects which profile they wish to play from a list of profiles that they have already created in the profile screen. When a profile is selected in the list its cell is highlight to indicate which profile was touched. Each profile in the profile list displays a unique name, current date of the profile and the total asset values of the profile.

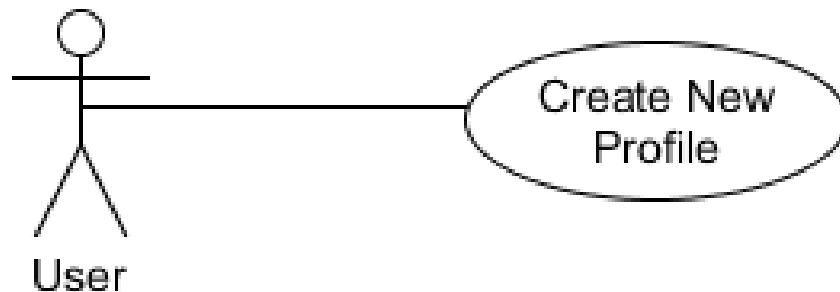
#### 4.1.3.2 Response Sequence

- 1) User navigates to the Profile Screen.
- 2) User scrolls through the profile list and finds the profile they wish to use.
- 3) User selects profile by touching its cell in the list.
- 4) User pushes the Select button start the simulation.

#### 4.1.3.3 Functional Requirements

REQ-1: Sqlite3 database which holds all profile information.

#### 4.1.4 User Case: Create New Profile



**Fig. 4.5** Create New Profile case.

#### 4.1.4.1 Description and Priority

Priority: High

Description: The user creates a new profile to run the stock simulation. A new profile can only be created if there are less than twenty one profiles in existence. The default name of the profile will always be profile\_x with the x representing the profiles iteration. The user can edit the profile but it must be unique. They can choose to have a random starting date or start on the first of any month of any valid year which are displayed in a list. If the user choose to have a random starting date then they can choose to have

completely random starting stocks, random stocks that are part of bubble, random stocks that are from a Bull market or random stocks that are from a Bear Market. If the user chooses their own starting date then they can only choose to have random starting stocks. The user can also choose not have any pre-purchased stocks and buy them when they start their simulation.

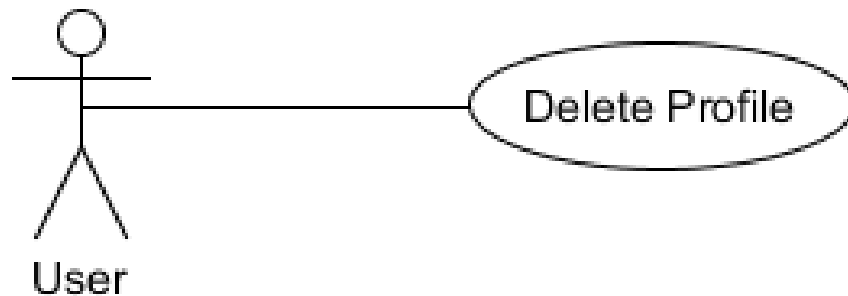
#### 4.1.4.2 Response Sequence

- 1) User navigates to the New Profile Screen by pushing the New button on the Profile Screen.
- 2) User either keeps the default profile name or edits it.
- 3) User either picks a starting date or selects random starting date.
- 4) User can then choose to have random stocks, bubble stocks, Bull stocks, Bear stocks or decide to choose which stocks they want when they start the simulation.
- 5) The user then pushes the create button and makes the new profile.

#### 4.1.4.3 Functional Requirements

REQ-1: Sqlite3 database which holds required stock information in order to pick stock for different scenarios as well as all valid dates.

#### 4.1.5 User Case: Delete Profile



**Fig. 4.6** Delete Profile case.

##### 4.1.5.1 Description and Priority

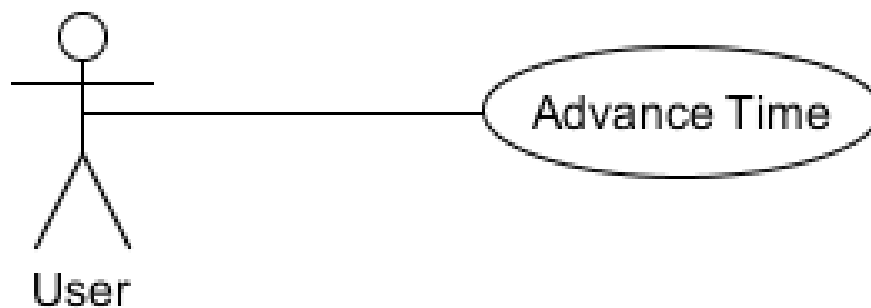
Priority: Medium

Description: This option allows a user to delete an old profile at any time in order to make room for new profiles.

#### 4.1.5.2 Response Sequence

- 1) User navigates to the Profile Screen.
- 2) User selects profile from the profile list.
- 3) User pushes delete button.
- 4) An alert prompt is displayed asking the user if they wish to proceed.
- 5) If user confirms their choice then the profile is deleted.

#### 4.1.6 User Case: Advance Time



**Fig. 4.7** Advance Time case.

##### 4.1.6.1 Description and Priority

Priority: High

Description: The function allows the user to advance the current time of the simulation to the next trading day. They can also choose to advance time in larger amounts by using a slider to select the number of trading days to wait. The maximum number of days that a user can advance time will be a year from their current date. When time is advanced the current price of all stocks they own are automatically updated. The user can only advance time to valid trading dates.

##### 4.1.6.2 Response Sequence

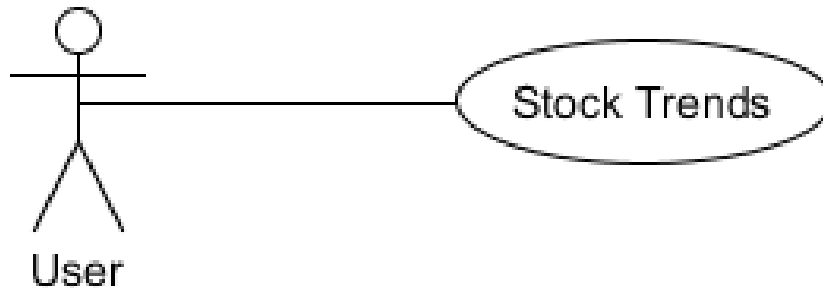
- 1) User navigates to the Portfolio Screen.
- 2) User selects the amount of time they wish to own.
- 3) User pushes the Advance button.

##### 4.1.6.3 Functional Requirements

REQ-1: Sqlite3 database which stores all valid trading dates.

REQ-2: System needs to retrieve stock data from Yahoo Finance.

#### 4.1.7 User Case: Stock Trends



**Fig. 4.8** Stock Trends case.

##### 4.1.7.1 Description and Priority

Priority: Medium

Description: Graph Stock Trends will display a simple graph of the stock price of a given stock. The user can choose to display the last 20 days or 3 months of trading of the stock. If the user owns any of the stock it will also display a break even line which represents the price of the stock at which the user has not lost money.

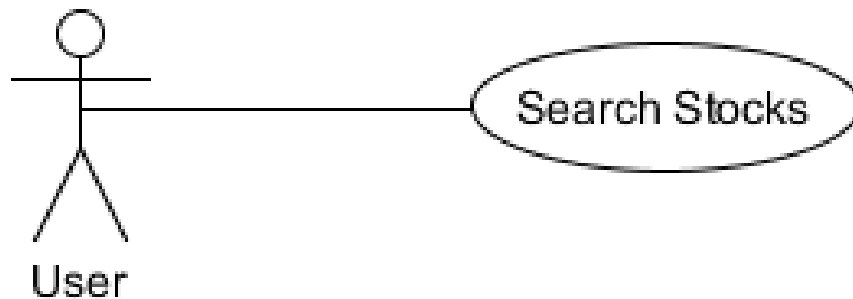
##### 4.1.7.2 Response Sequence

- 1) The user needs to navigate to the Portfolio, Buy or Sell Screen.
- 2) In the stock list the user pushes the Trends button next to the stock of interest.
- 2) On the Trends Screen the user then pushes the 20 days or 3 months button.

##### 4.1.7.3 Functional Requirements

REQ-1: System needs to retrieve stock data from Yahoo Finance.

#### 4.1.8 User Case: Search Stocks



**Fig. 4.9** Search Stocks case.

##### 4.1.8.1 Description and Priority

Priority: High

Description: This feature allows the user to search for stocks by their stock symbol. The only stocks they can search for are stored in on the phone in a database. If a user searches for “g” the search results will display all stock symbols that begin with the letter g. The current price and the amount owned of each stock in the search result are displayed. Only twenty five stocks are shown at a time to limit the amount of stock prices that need to be retrieved from Yahoo Finance.

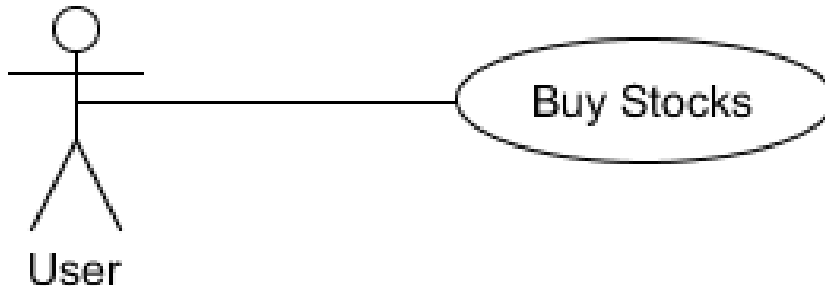
##### 4.1.8.2 Response Sequence

- 1) User navigates to the Buy Screen.
- 2) The user enters their search query.
- 3) User pushes the Search button.
- 4) Search results are displayed.

##### 4.1.8.3 Functional Requirements

- REQ-1: System needs to retrieve stock data from Yahoo Finance.  
REQ-2: Database needs to store all valid stock symbols.

#### 4.1.9 User Case: Buy Stocks



**Fig. 4.10** Buy Stocks layout.

##### 4.1.9.1 Description and Priority

**Priority:** High

**Description:** This feature allows the user to use money in their bank account to buy stocks. They can only purchase as much shares at a time as they can afford. The user can only purchase shares in one stock at a time. They are also limited to owning 10 stocks at a time. Before the purchase is completed an alert message is shown to ensure that the user want to make the purchase.

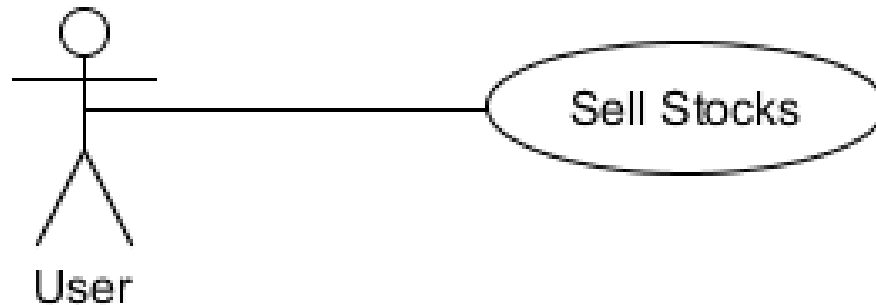
##### 4.1.9.2 Response Sequence

- 1) User navigates to the Buy Screen.
- 2) User searches for stocks and selects stock they want to buy.
- 3) User selects amount of the stock they want to buy.
- 4) User pushes Buy button.
- 5) An alert message is shown asking the user if they are sure they want to make the purchase.
- 6) If the user proceeds with the purchase the cost is withdrawn from their bank account and the stock is added to their portfolio.

##### 4.1.9.3 Functional Requirements

- REQ-1: System needs to retrieve stock data from Yahoo Finance.  
REQ-2: Database needs to store all valid stock symbols.

#### 4.1.10 User Case: Help



**Fig. 4.11** Sell Stocks case.

##### 4.1.10.1 Description and Priority

**Priority:** High

**Description:** This feature allows a user to sell stocks that are currently in their portfolio. They can choose to sell as many shares of any of the stocks that they own at any given time. Before the sale is complete the user is prompted to confirm the transaction.

##### 4.1.10.2 Response Sequence

- 1) User navigates to the Sell Screen.
- 2) User selects the stock they want to sell.
- 3) The user selects the amount of stock they want to sell.
- 4) User pushes the sell button.
- 5) An alert message is shown asking the user if they are sure they want to sell the stock.
- 6) If the user decides to sell their stock the money is deposited in their bank account.

##### 4.1.10.3 Functional Requirements

REQ-1: System needs to retrieve stock data from Yahoo Finance.



## 4.2 Profile Functional Requirements

### 4.2.1 Limit on number of different stocks owned

**Priority:** Medium

**Description:** A profile can own no more than 10 different stocks at a time. There is no cap on the number of shares of a stock that the profile can have, thus the user can buy shares of a given stock if they are at the maximum number but already own the stock. If the user is at the maximum number of stocks, selling all shares of given stock will allow them to purchase a new stock.

### 4.2.2 Limit number of profiles that can exist at a time

**Priority:** Low

**Description:** The maximum number of profiles that a user can have created at a time will be limited to 20. If the maximum number of profiles has been reached, deleting one of the profiles will allow the user to make another one.

### 4.2.3 Profile must have unique identifier

**Priority:** High

**Description:** All profile must have a unique name when they are created. This rule must be enforced whenever a new profile is created.

### 4.2.4 Profile must store its current time

**Priority:** High

**Description:** All profiles must keep track of their current time while the user is playing the game and after they have stopped.

### 4.2.5 Profile must store all data on currently owned stocks

**Priority:** High

**Description:** The profile has to store which stocks the user currently owns along with the number of shares that the user owns and the current price of the stock in relation to the current time of the profile.

### 4.2.6 Profile must store bank account value

**Priority:** High

**Description:** The profile must store the amount of money that the user has in their bank account.

## 4.3 Sqlite3 Functional Requirements

### 4.3.1 Store all valid stock symbols and related data

**Priority:** High

**Description:** The database must store a valid stock symbols that a user can search for and buy. It must also store the date that the stock first became available to purchase and, if applicable, the date it was no longer available to purchase.

### 4.3.2 Store all valid trading dates

**Priority:** High

**Description:** The database must store all valid trading dates that the stock exchange is open and assign them a descending numeric value. This will allow time to be incremented by simply subtracting an integer from the current in order to advance time.

### 4.3.3 Store list of Bear Market Dates

**Priority:** Medium

**Description:** The database must store a list of dates for when Bear market was occurring.

### 4.3.4 Store list of Bull Market Dates

**Priority:** Medium

**Description:** The database must store a list of dates for when a Bull Market was occurring.

### 4.3.5 Store list of Bubble Stocks

**Priority:** Medium

**Description:** The database must store a list of stocks that were part of a bubble at one point in time that can be grouped together by date.

### 4.3.6 Store profile data

**Priority:** High

**Description:** The database must store all data related to a user profile at any given time.

## 4.4 Other Functional Requirements

### 4.4.1 Update profile when time is advance

**Priority:** High

**Description:** The users profile has to be updated each time the current date in the simulation is advanced. This means all the prices of every stock they own and the current date of the profile need to be changed accordingly.

### 4.4.2 Limit maximum amount of stock user can buy

**Priority:** High

**Description:** The maximum amount of stock that a user can buy needs to be limited by the total amount of money they have in their bank account. They can never have the option to purchase more stocks then they can afford.

### 4.4.3 Limit maximum amount of stock user can sell

**Priority:** High

**Description:** The maximum amount of a stock that a user can sell has to equal the maximum amount of the given stock that the user owns.

### 4.4.4 Colors to indicate whether a stock increased or decreased

**Priority:** Low

**Description:** If the opening value of a stock is larger than the closing value the price of the stock should be displayed in red. If closing value is larger than the opening value its price should be displayed in green. If both values are the same than the stock price should be shown in black.

### 4.4.5 Limit number of stock displayed from search

**Priority:** High

**Description:** The number of stocks displayed to a user at a time should be no greater 25.

### 4.4.6 Limit loading stock data

**Priority:** High

**Description:** Stock data should only be loaded for stock that are being currently displayed or are in the user's profile.

#### **4.4.7 Color Blind Mode**

**Priority:** Low

**Description:** The color settings of the application can be changed so a person with red-green color blindness can distinguish between the two.

#### **4.4.8 Retrieve Data from Yahoo Finance**

**Priority:** High

**Description:** The application needs to be able to retrieve stock data from Yahoo Finance as the user advances time and searches for stocks.

## **5. Other Nonfunctional Requirements**

### **5.1 Performance Requirements**

Since the application may be making multiple queries to an external database in order to retrieve stock data, the user may have to wait a considerable amount of time. These wait times should be minimized by getting stock data for large blocks of time to avoid having to get data for every stock in the user's portfolio when the user advances time.

### **5.2 Safety Requirements**

Bazhanga does not access or modify any files or data stored on the user's phone that are not part of the application; therefore, it can never unintentionally delete or corrupt any user files or data. It will never put the user's phone at risk of any kind of mechanical damage since it will not require any extraneous physical activity to use. The only safety requirements that apply to the operation of this application also apply to using any application or smart phone device; i.e. using while operating a motor vehicle.

### **5.3 Security Requirements**

Since Bazhanga will not require any personal information to run, it can never compromise any of the user's personal information. The application itself has no password protection and can be accessed by anyone using a phone with it installed. It assumes that anyone accessing the application will have the user's consent.

### **5.4 Software Quality Attributes**

#### **5.4.1 Graphical User Interface**

The design of Bazhanga's graphical user interface is to be based on usability as well as having a visually appealing layout. At no point should the user feel like they cannot navigate through the application due to an overly complicated interface. Adequate feedback should always be provided to the user for every event that occurs that will affect the user's experience. For example, if the user has to wait for data to be loaded from an external database, they should be informed of the reason for the pause.

### **5.4.2 Data Connection**

The user should also be able to access the application whether or not they have an internet connection. Since the stock data is retrieved from an external database through the internet, a user will not be able to update the current day of their portfolio if they do not have an internet connection. They should, however, still be able to access their portfolio and view which stocks they own and their current value.

## **5.5 Business Rules**

The user can access the application at any time. The user also has complete control of when the application advances in time and over what stocks they want to buy and when they want to sell them.

## 6. Other Requirements

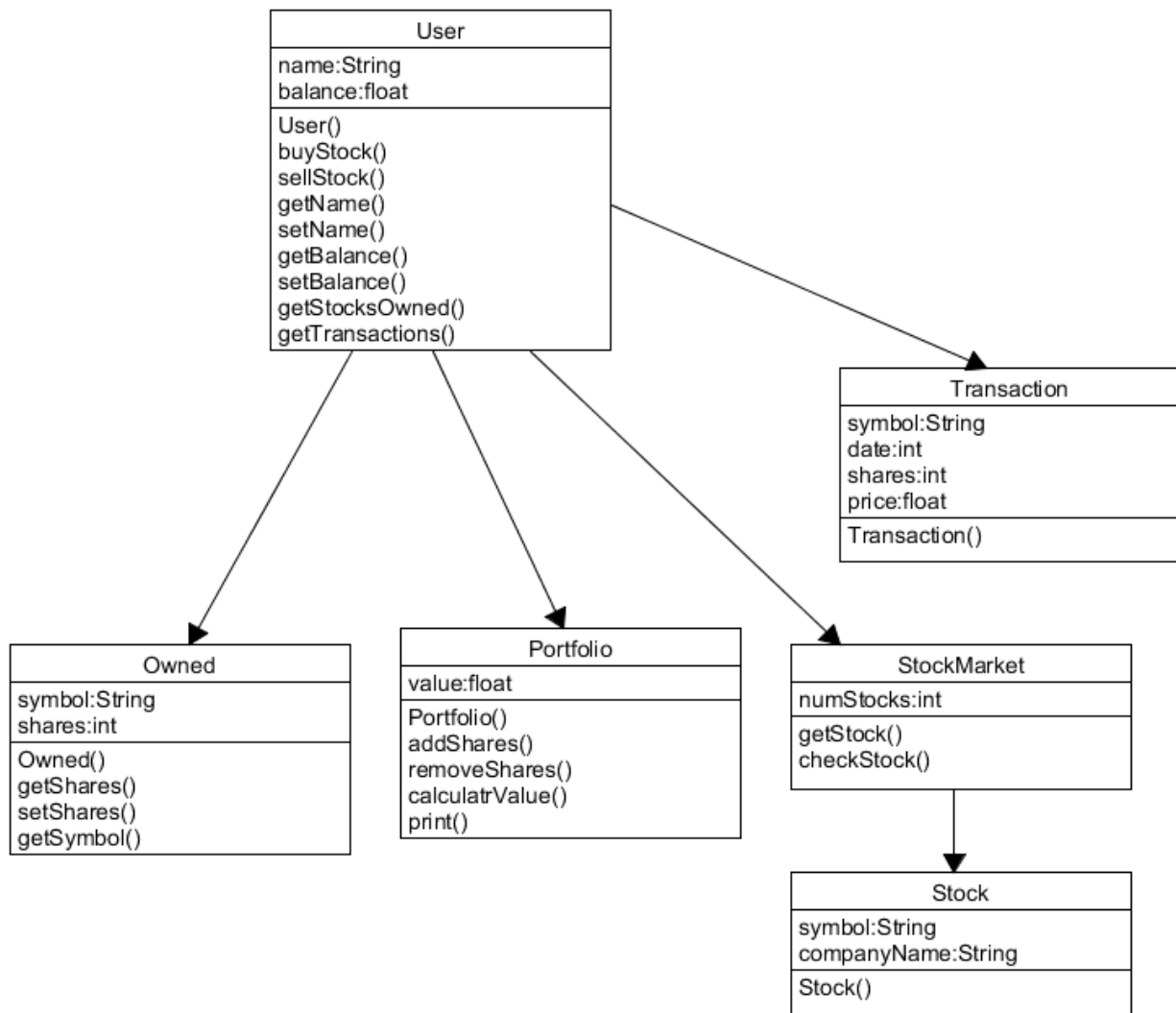
### Appendix A: Glossary

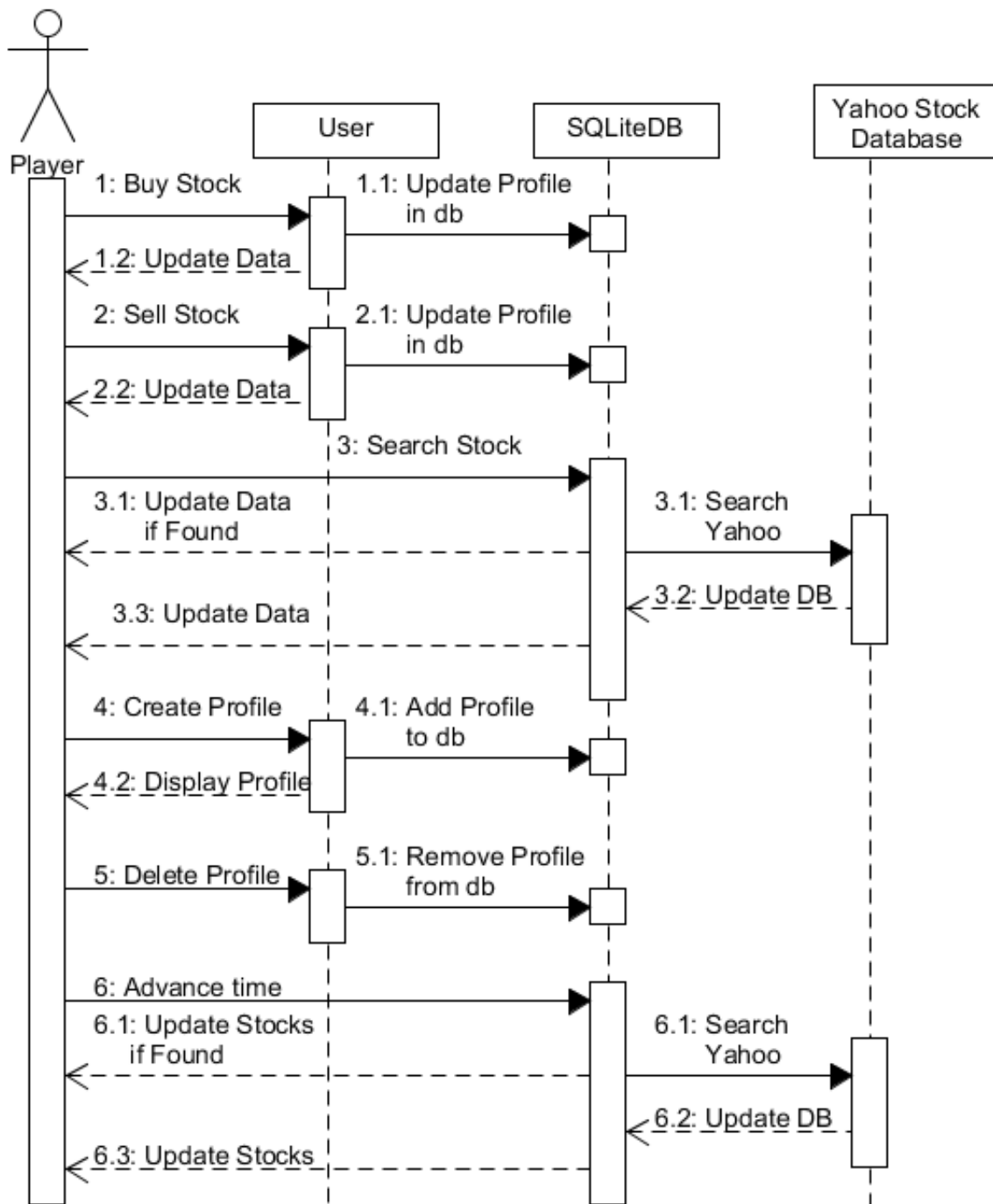
Term	Definition
<b>FMDB</b>	An Objective-C wrapper built around SQLite.
<b>Portfolio</b>	In finance, a portfolio is a set of investments that are owned by a person or organization.
<b>Swift</b>	A programming language for Cocoa and Cocoa Touch application.
<b>X-code</b>	An integrated development environment with software development tools for making software for Apple's OS X and iOS.
<b>Yahoo! Finance API</b>	Stock database managed by Yahoo that contains current as well as past stock data.
<b>User</b>	Person using the application.
<b>Bear Market</b>	The average value of the stock market is moving in a downward direction.
<b>Bull Market</b>	The average value of the stock market is moving in an upward direction.
<b>Stock Bubble</b>	When the value of a stock or stock is driven above what they would normally be worth.
<b>Color Blind</b>	Inability for a person to perceive the differences between some colors.
<b>Milestone</b>	Event marking a significant stage in development.

<b>Bazhanga</b>	An exciting new Stock Market simulation application meant to teach people about owning stocks.
<b>Case Diagram</b>	A diagram representing a user's interaction with a system that shows the relationship between the user different use cases.
<b>Functional Requirement</b>	Defines a function of a system and its components.
<b>Non-Functional Requirement</b>	A requirement that specifies conditions that can be used to determine the operation of a system.



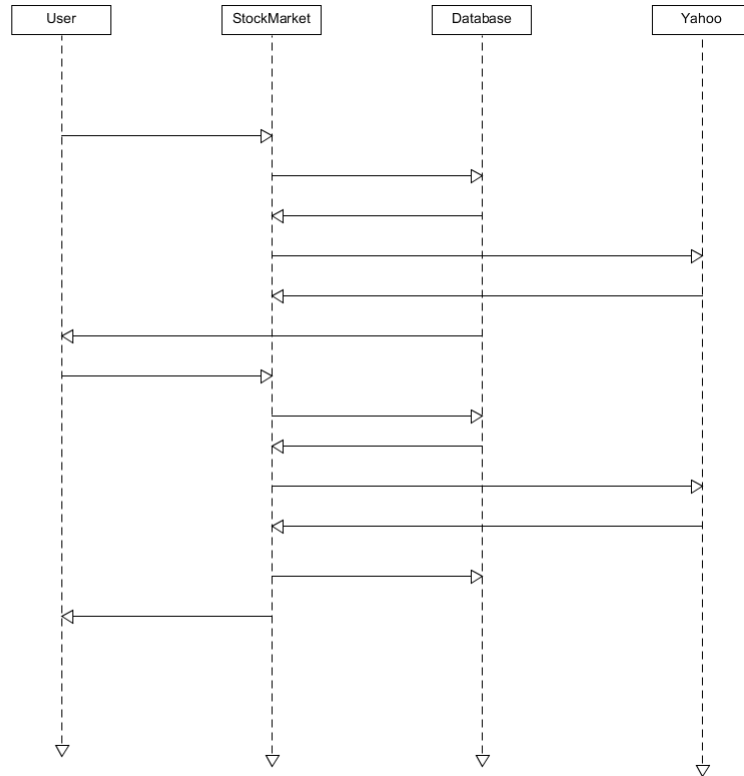
## Appendix B: Analysis Models





User wants to buy a stock but doesn't know the stock symbol for Microsoft.

1. User sends a search to the StockMarket class for "Micro".
  2. The StockMarket class queries the database for company name.
  3. No companies named "Micro" are in the database.
  4. StockMarket makes a query to Yahoo for stocks named "Micro."
  5. Yahoo responds to the StockMarket with a list of stocks.
  6. StockMarket sends the list to the User.
  7. User sends the symbol of the stock symbol "MSFT" to the StockMarket class.
  8. StockMarket queries the database for MSFT.
  9. The Database does not have MSFT.
  10. The StockMarket queries yahoo for MSFT.
  11. Yahoo responds with information about the stock.
  12. StockMarket stores the data to the database.
  13. StockMarket returns the Stock MSFT to the User.
- MSFT is then loaded as the current stock.



## Appendix C: Milestones

Event	Data	Description
First Prototype	01/29	Complete first prototype of the application in Xcode.
First Draft of SRS	02/05	Finish the first draft of the Software Requirements Document.
Revise Prototype	02/12	Revise the prototype to better reflect the final product .
Second Draft of SRS	02/24	Revise the Software Requirements Document to better reflect the new prototype.
Implement Core Features	02/31	Fully implement all of the core function of the application and test on iPhone.
Implement Graphing Features	03/7	Add the graphing features to the application.
Test Application	03/14	Extensively test the application and remove any bugs that are found.
Complete Final SRS	04/09	Finish the SRS for the application.