# Stock Simulator

Requirements

1. **Introduction and Context (update from document 1 section 1 and 2)**

The product will allow users to simulate stock trades through a Broker. Each simulation will allow for a different, possibly random set of market values to be 'played' in the Market Simulator and the user can make trades at an accelerated rate. The rate at which the market advances will be increased from the historical changes. As each session progresses, a User will be able to see how their trading performance ranks against other users playing who are also playing the simulation.

1. **Actors and their Goals**

The users of the product will interact with the broker and stock server through a gui client. The goal of the actors will be to practice interacting within a market environment as well as compete against other users.

1. **Interacting processes**

**Stock Data Server**

* Stores a set of chronologically-ordered stock value data
* Broadcasts the next update of stock values to clients and broker via multicast
* Provides a stock history when requested by the User

**Market Simulator (Broker)**

* Listens to the Stock Data Server for stock price updates
* Receives buy/sell requests from Users
* Validates a user’s requested trade against their current assets (per the Portfolio Manager) as well as the stock price that they have requested (via their recently known stock values received from the Stock Data Server)
* Maintains a running leaderboard that lists the most valuable portfolios in descending order and serves this list to all clients as multicast

**Trade Simulator (Client)**

* Provides client UI for the simulation
* Listens to the Stock Data Server for stock price updates
* Makes registration/authentication/buy/sell requests to the Broker
* Listens for updated Leaderboard as broadcasted by Broker

**Portfolio Manager**

* Stores user credentials
* Stores user accounts (cash/stock)
* Tracks a user’s past trades
* Serves user account info to Broker when requested

1. **Functional Requirements**

* **A new user needs to be able to register**
  + A username cannot already exist in portfolios.
  + The password must be typed twice to avoid typos.
  + Once user input is validated, a new portfolio must be created, persisted, and protected with the password.
* **A user needs to be able to authenticate their access to recieve portfolio data**
  + A user will provide a username and password to gain access to an existing portfolio
  + For ease of development, a User Client process may cache the the user’s credentials for future use
* **A user needs to be able to see current stock data**
  + Once a user has authenticated, their client will begin listening for Stock Price Updates from the Stock Data Server.
* **A user needs to be able to buy/sell stock**
  + A user must be able to request the trade of a stock from the Broker.
  + A user must receive a confirmation or failure message for each trade request.
  + Upon a successful trade, the user’s portfolio must be updated in both the Portfolio Manager and the user’s screen.
* **A user needs to be able to see a leaderboard**
  + A user must be able to see the most recent leaderboard that has been broadcast by the Broker.
* **A user needs to be able to see the recent history of a stock**
  + A user must be able to request a past history of all or a given stock from the Stock Data Server.

1. **Non-functional Requirements**

* Developed in C# using Microsoft Visual Studio 2015 under a student license
* All work artifacts are managed through git
* Use Trello as a scrum board to manage workflow

1. **Future Features**

* User-defined length of stock history reports
* User session tracking/management (MVP has minimal/stateless “authentication” system)
* Mutating stock progression (so a single set of stock data can repeat but mutate, avoiding visible patterns)

1. **Glossary**

**Stock Data Server:** The process responsible for storing and providing, at regular intervals, stock data updates.

**Broker:** The process which handles all user requests except for a stock history request.

**Portfolio Manager:** The process in which a user’s portfolio credentials, cash, and stock information are stored.

**Client:** (Trade Simulator) The user interface providing entry points for all user interactions and requests.