# Stock Simulator

Architectural Design

1. **Introduction and Context**

The current thought is to allow users to simulate stock trades through the 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. **Users 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)**

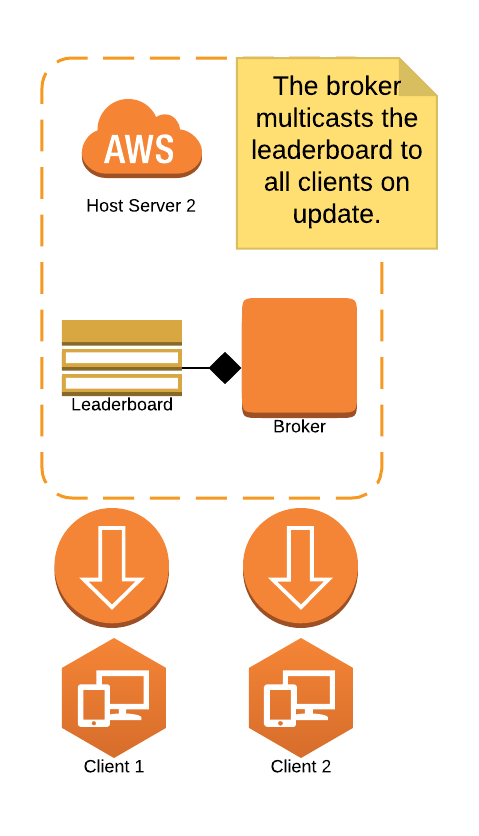
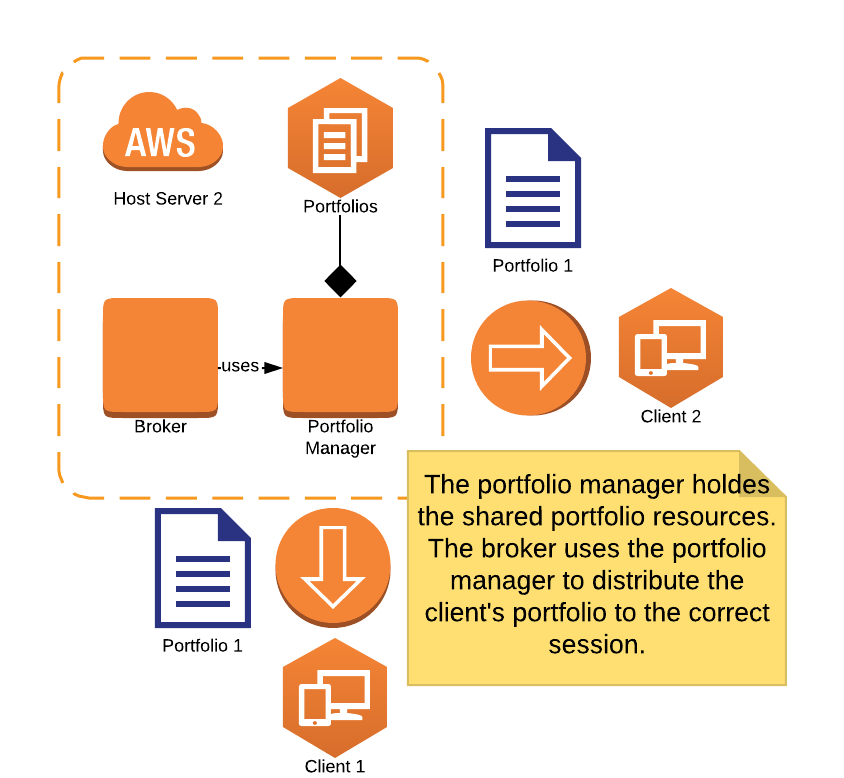
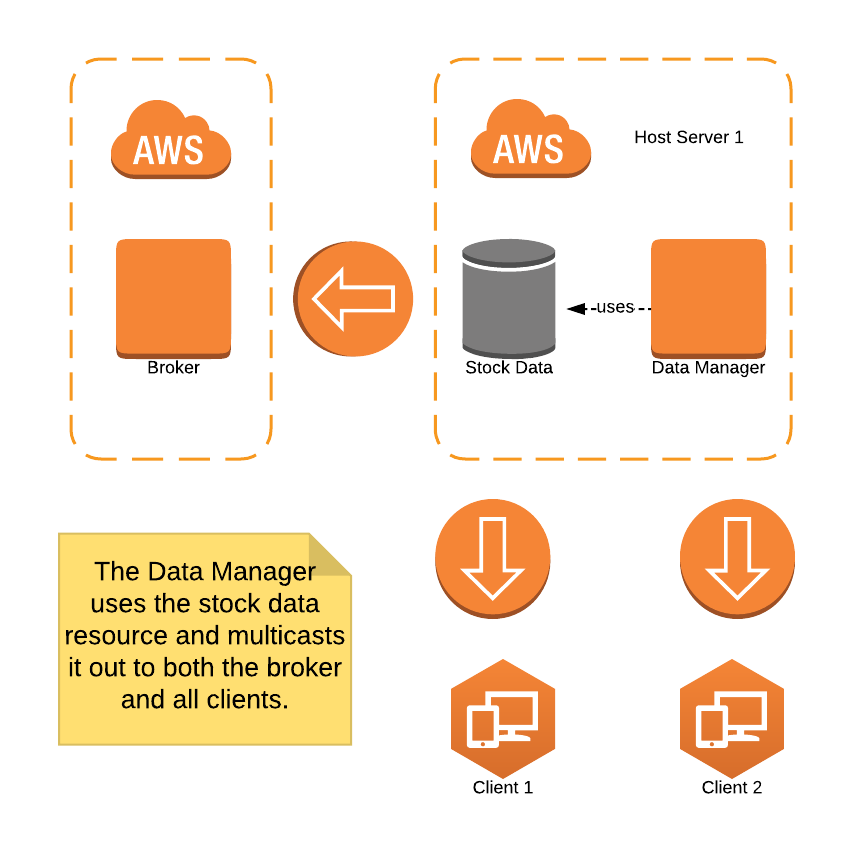
* Listens to the Stock Data Server for stock price updates
* Receives buy/sell requests from Users
* Validates a user’s requested trade against their current assets (per the Portfolio Manager) as well as the stock price that they have requested (via their recently known stock values received from the 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**

1. Stores user credentials
2. Stores user accounts (cash/stock)
3. Tracks a user’s past trades
4. Serves user account info to Broker when requested
5. **Major Components and Their Relationship**

****