COMP4920 Project Documentation

Project: Educational Stock Market Simulation Game

Group: Taiyun Kim
Andrea Olrich
Jack Walsh
Sam Bassett
John-Paul Meyer

Tutorial: Wednesday 1 - 3pm

Tutor: Robert Clifton-Everest

Table of Contents:

1.0 Project Overview

Our project is a single-page web-based application that allows users to register a unique account through which they can trade stocks on a simulated system for free. The system is aimed at being realistic while also educational.

1.1 Main Views

The application has six main views:

1. **Home** - this page is displayed if the user is not authenticated with the system. It explains various features of the product/system and options to Login/Register.

2. **Welcome** - the ‘Home’ page for authenticated users which describes the four main functions available.

3. **Trade** - allows users to select stocks from the ASX and see real-life data regarding company performance before opting to buy/sell the stock.
    a. Pressing ‘Place Order’ opens a modal to select the number of stocks and order type. This order is then confirmed before being placed.

4. **Portfolio** - displays a summary of the user’s performance, currently held stocks, and pending stock orders. Users can cancel orders or sell stocks.

5. **History** - displays information about previous transactions, including net gain/loss on stocks sold.

6. **Leaderboard** - shows a list of all users in the system ranked by their total portfolio value.

2.0 Technologies Used

The following technologies are used in the application:

- Google App Engine is used to host the application (at https://risk-4920.appspot.com/) and as a RESTful server for User and Stock data (see *3.0 System Design* for more details). This is written in Python.
    - Cron is used to schedule the execution of stock orders every minute.
    - Memcache is used to store data from the Yahoo Finance API for optimisation.

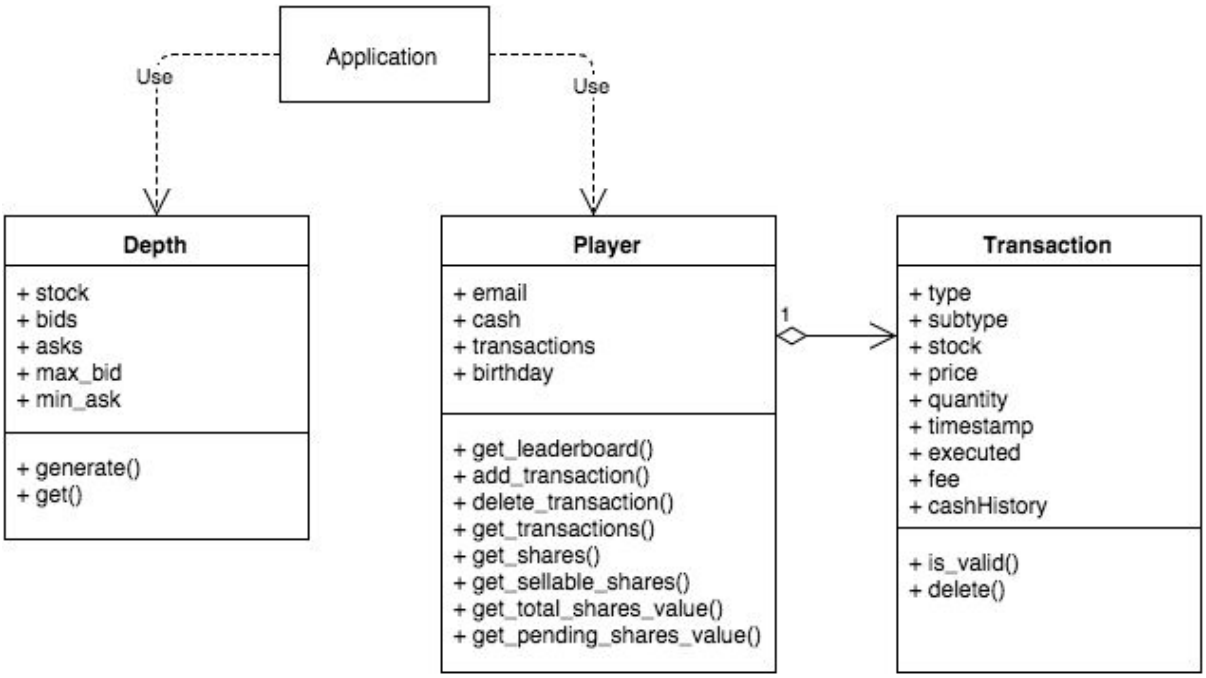
- AngularJS is used as an MVC (Model-View-Controller) framework to control the various views of the Single Page Application (SPA). These views are described in *1.1 Main Views*.

- Twitter Bootstrap is used as the basis of the UI/UX.

- HighCharts is used for the interactive javascript charts.

3.0 System Design

3.1 Basic UML of Data Structure



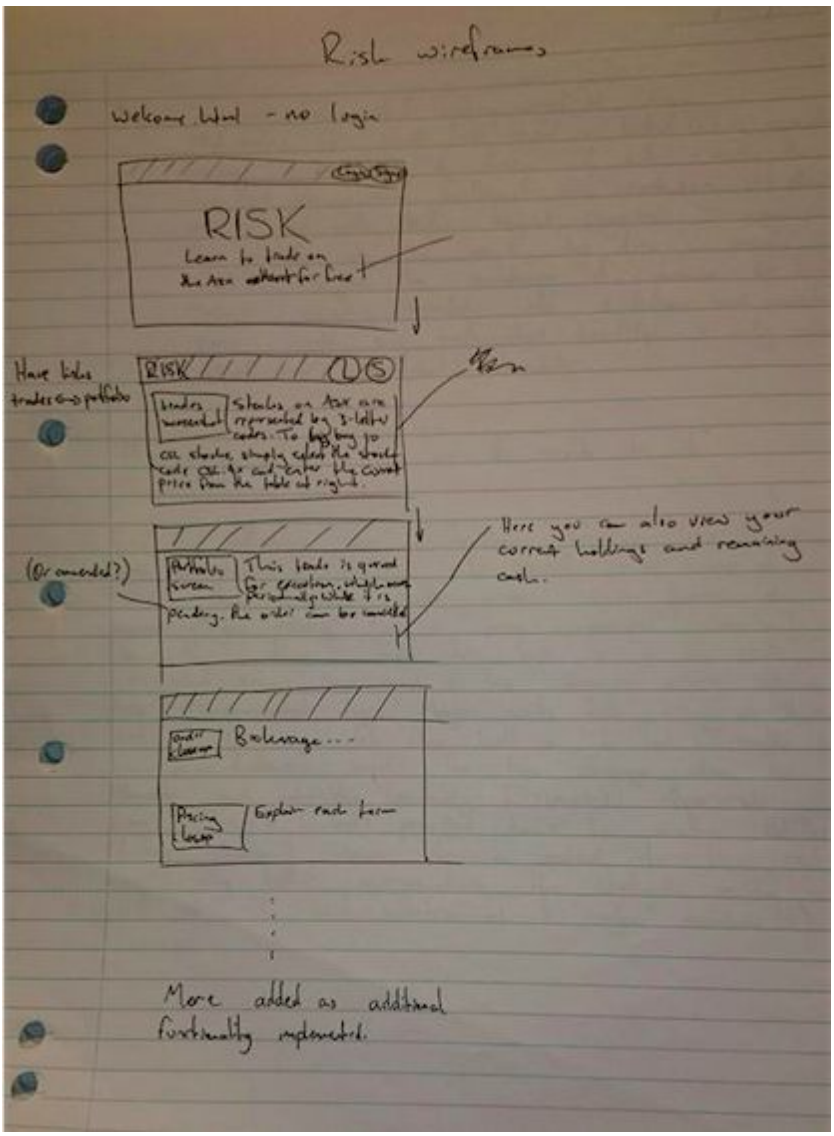
3.2 RESTful API Structure

A path exists for each entity that allows for Create Read Upload Delete (CRUD) operations (or a subset of these).

/account	Handles account authentication (login, signup, logout, etc.)
/order	Handles orders/transactions made in the system related to the Transaction class.
/depth	Handles generation of simulated depth information for market stocks through the Depth class. All other information is fetched from an external API.
/user	Handles requests for user information and manipulations of the Player class.

4.0 UI Mockups

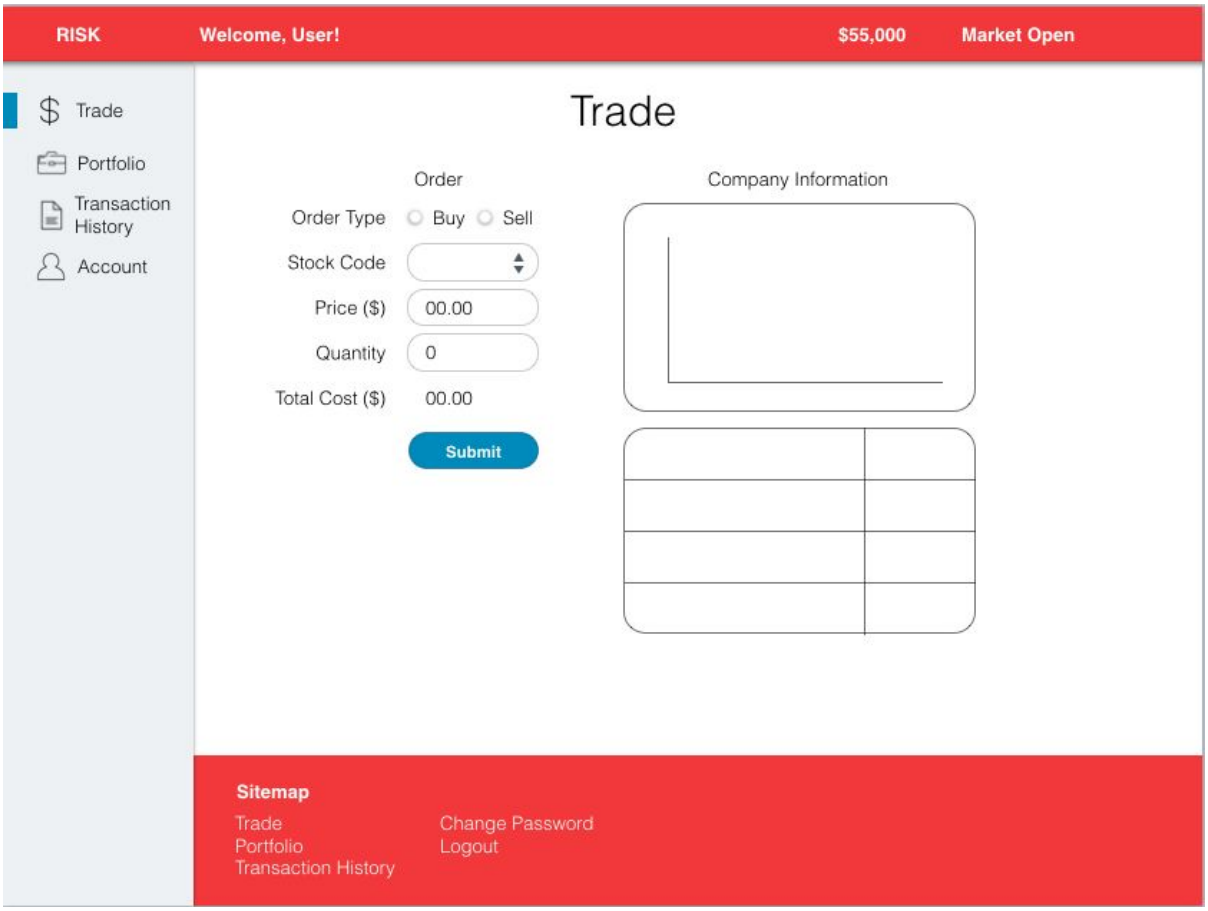
4.1 Welcome Page Mocks



4.2 Final Design Mocks

Trade Page:

- Consistent navigation (top bar and side bar) across all pages.
    - Ability to access all views from all other views (not a centralised home page)
- Introduction to the new colour scheme



Example of a modal (to be used for all account-related forms - login, register, change password).