CS 157A Intro to Database Management Systems

Project Proposal

Project Title: Stock Data Aggregation Web-App

TEAM 34

Sachin Shah

Yang Li

En-Ping Shih

Professor: Dr. Mike Wu

TA: Sriram Priyatham Siram

San Jose State University

September 6th, 2019

Project Overview

Our project will be a web-based application which can handle user/client input and report back to the user all relevant information regarding a certain company’s stock. The application will provide up-to-date, useful information on the security of the client’s choice and also recommend to the client whether they should purchase the security in question. The application will do this by first retrieving information from various online sources for financial information. Then the application will perform operations using the retrieved data to determine if the security should be purchased by the user. For example, retrieving the beta value, which is a measure of a securities volatility, could be compared to a benchmark value. Based on this comparison, the security could be assigned a Boolean value of “buy” or “sell.” Another function of our application would be storing the results of client queries in a database which will be hosted on a remote server.

The primary stakeholders for this application are people who may not have extensive knowledge of the stock market and are looking for a way to get started with investing. Furthermore, these are people who are interested in investing in public securities and not within the private sector, as a lot of the information on private companies is difficult to come by online. Our stakeholders value time and convenience when it comes to retrieving reliable information about publicly traded securities. Since this is a web based application and it meeting the functional requirements requires it to be used by end-users, this stakeholder group outlined above is of the upmost importance. Our application is important because nowadays speculators and passive investors are looking for a way to get a buy or sell decision without having to do extensive research. Our site will retrieve information and not require the user to do anything other than enter the ticker name of a security. The user saves time, energy, and resources while receiving a user-oriented service.

System Environment

To build this application successfully, we will need to set up the environment based on the three-tier architecture (Figure 1) which contains the client, the server, and the database. On the client part, we have to make sure our web-based application is working on internet browsers on any computer or laptops for our users and clients. (ex. Google Chrome and Safari), and JavaScript will be our frontend development language. Apache will be the HTTP server that we are going to set up on our laptop/PC and will be using python and XML to develop. At last, We will use JDBC as our data access API, which is in Java.

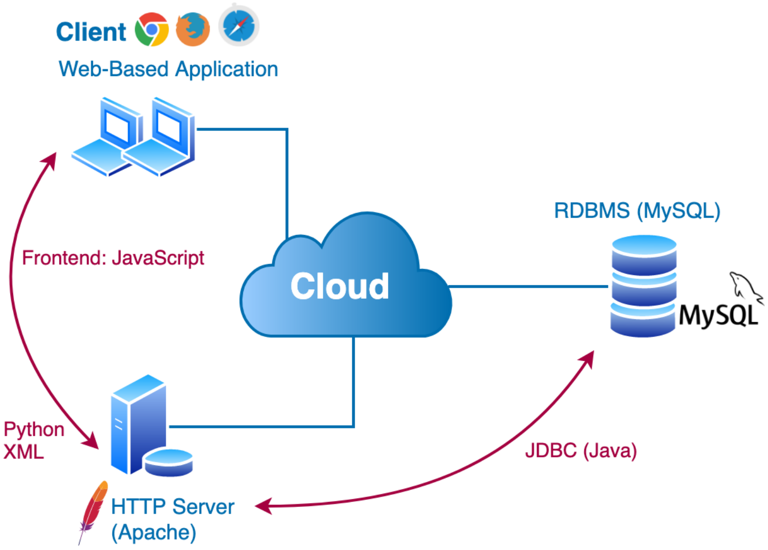

Figure 1. Three-Tier Architecture

Functional Requirements

- Users' should be able to sign up, log in and log out their accounts through the web application.
- Users' should be able to see their search queries, by typing the company name or the stock name.
- Users' should be able to store selected search result to their personal watchlist as "Stocks Watchlist".
- Users' should be able to use the personal watchlist to update their favorite stocks or company, or they can delete them if the stocks are not their favorites anymore.
- Users' profile should list the recent searched stocks, the favorites stocks, as well as the most potential stocks.

Non-Functional Requirements

Front-end: React

Server: Apache

Database: using MySQL and JDBC, to store stocks information and personal stocks watchlists.

Access control and Security: using Google Identity Platform, using firebase to store user account information such as username and password.

A second non-functional requirement is an issue regarding the security of user data. When our users create a new account, their information, such as user id, password, and email need to be stored in a secure location. To address this issue, we plan on using the Google Sign-in API via the Google Identity platform, a third-party service which allows developers to incorporate a secure way for users to use their existing google accounts to log in to our web-app.

In terms of access to the database storing users' stock watchlists, access control will be handles in the logic layer of our application. Once a user is logged in, they can edit and generate a view of the table associated with their unique user ID. They will only have access to their own watchlist.