**Twitter Fingers**

Requirements

Julian Mathis (PM)

Dorian Wood

Anisha Carter

Taylor Clark

Dominique Collins

Sydney Parker



Contents

[1.0 Introduction 3](#_Toc481445123)

[*1.1 Goals and Objectives* 3](#_Toc481445124)

[*1.2 Statement of Scope* 4](#_Toc481445125)

[*1.3 Software Context* 4](#_Toc481445126)

[*1.4 Major Constraints* 5](#_Toc481445127)

[2.0 Usage Scenario 6](#_Toc481445128)

[*2.1 Use-cases* 6](#_Toc481445129)

[*2.2 Special Usage Considerations* 6](#_Toc481445130)

[3.0 Data Model and Description 6](#_Toc481445131)

[4.0 Functional Model and Description 8](#_Toc481445132)

[*4.1 Description of Programs* 9](#_Toc481445133)

[*4.2 Design Constraints* 10](#_Toc481445134)

[*4.3 Software Interface Description* 11](#_Toc481445135)

[*4.3.1 Human interface* 12](#_Toc481445136)

[5.0 Behavioral Model and Description 12](#_Toc481445137)

[*5.1 Description for software behavior* 12](#_Toc481445138)

[*5.1.1 Events* 13](#_Toc481445139)

[6.0 Restrictions, Limitations, and Constraints 14](#_Toc481445140)

[7.0 Version History 14](#_Toc481445141)

# **Introduction**

Twitter Fingers has been tasked with creating an application that will deliver the latest finance, stock, and investment information in real-time. This document outlines in detail what requirements this project will have, as well as the requirements Twitter Fingers has decided we will fully develop in order to deliver this product to the client so that meets their needs.

## 1.1 Goals and Objectives

Our goal for this project is to construct an application/program that will pull finance and stock market related tweets from Twitter, and display the information to the user in a constantly updating screen. These tweets will be able to be identified according to various identifiers such as the stock symbol of the related company/business. The tweets will come from a list of Twitter accounts that the user will get to choose. The tweets will be displayed in real time, meaning that as these pre-approved Twitter accounts tweet, they will be displayed to the user through our application. This means the application will continuously update based on the influx of live tweets. Once the tweets have been displayed, the user will be able to stop the influx of tweets and choose which tweets they want to currently display based on the information contained in the tweets. All of the incoming tweets will be stored in a database so that they can be referenced later on. Our application will also analyze the tweets and provide the user with statistics regarding the tweets or the information within the tweets. The purpose of this project is to make users aware of current events in the financial and stock market world as they are happening, so that investment decisions can be made effortlessly and swiftly.

## *1.2 Statement of Scope*

The team has been given an outline of task they must complete.

* Pull tweets in real-time using a Twitter API
* Plan and program a database that holds information within tweets
* Display the tweets to the screen in the form of a web application
* Store tweets in the database
* Have this stored financial information be accessible by various queries.
* Create a fully interactive interface
* Analyze our data to provide statistics to the user

Twitter Fingers has been tasked with creating a web application that reads in tweets that contains financial information from Twitter, displays the tweets in a scrolling manner, and stores them in a NoSQL database so that they can be displayed again later. The information from the tweets will be pulled using a Twitter API. Specifically, we will use the Twitter Streaming API to collect these tweets display them to the user.

## *1.3 Software Context*

|  |  |  |
| --- | --- | --- |
| **Software** | **Description** | **Type** |
| MongoDB | MongoDB is a free and open-source cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schemas. | Database |
| Google Drive | Google Drive is a cloud storage service that lets users store and synchronize digital content across computers, laptops and mobile devices | File Sharing |
| GitHub | GitHub is a web-based Git or version control repository and Internet hosting service. | Version Control |
| Twitter Streaming API | The Twitter Streaming API allows high-throughput near-realtime access to various subsets of public and protected Twitter data. | API |
| XAMPP | Web servers allowing script to be run in a variety of programming languages. We will run a local server on our computers for testing purposes. | Web Server |

## 

## *1.4 Major Constraints*

**Database** - Creating a database that will handle an influx of data from tweets may be a difficult task.

**Implementation** – Learning both how to maintain a database and work with the Twitter APIs.

# **2.0 Usage Scenario**

The purpose of the program is to have financial data be easily displayed to the user. The user will be able to be well informed on news in the financial and stock market field. Tweets will also be analyzed in order to provide more useful information about the tweets. This tool will make the user more financially aware.

## *2.1 Use-cases*

The basic use of this software is for the user to have easy access to the latest financial news via a web application.

## *2.2 Special Usage Considerations*

The overall usage consideration is for this project to be able to pull financial and stock market related tweets from Twitter, display the information to the user in real time, and analyze the tweets to provide statistics to the user. This has potential to be used in a real stock market, seeing as how it can provide finance and investing news quickly.

# **3.0 Data Model and Description**

**Web-Based Platform**

This program is a web-based platform that will be visible on a desktop executable program or website. When the program is launched, the user will be able to start a stream of tweets as they are being tweeted in real time. The financial information will then be stored to a database, so that the user may view tweets that have already been tweeted that they may have missed. The application will allow the user to choose specifically what type of information from the tweets they want to see. This will be done by allowing the user to filter the data that will query the database for their desired information. The information that they will want to search will be related to a specific financial topic. The program will be coded using HTML, CSS, PHP, and JavaScript. A web server will be required to connect to the API.

**Database**

The database that will be used for this project is known as MongoDB. MongoDB is an open source, document-oriented database designed with both scalability and developer agility in mind. Instead of storing your data in tables and rows as you would with a relational database, in MongoDB you store JSON-like documents with dynamic schemas. This section describes the potential advantages and disadvantages that come with using this software tool.

**Advantages:**

* **Simple Implementation:** MongoDB is relatively easy to install on both Windows and Mac, and setting up the environment is relatively simple
* **Cost Efficiency**: MongoDB is an open source database tool, making it widely accessible for all members of the team to contribute with management of the database
* **Compatibility**: MongoDB is consistently being updated and is compatible with many programming languages that are used today

**Disadvantages:**

* **Limitations**: MongoDB can scale horizontally, meaning we can add more servers should we require more space for the information we receive. This uses lots of connections to do so and it requires managing lots of special nodes
* **Scalability**: Data size in MongoDB is typically higher due to the fact that each document has field names stored and less flexibility with querying (e.g. no JOINs are searchable)

# **4.0 Functional Model and Description**

**NoSQL Database**

A NoSQL database will be used and will store financial, stock, and investment information from the displayed tweets. The database will be used to organize incoming information by translating into a more readable and organized format; The tweets will be organized by the company names, stock prices, investment recommendation, stock symbols and stock trends. You will be able to search any of these features to retrieve information. With our information organized in a database, it will allow for the similar data to be referenced from the database and displayed together. We will be using MongoDB to build our database. MongoDB allows stores the tweets in a JSON like format, which is useful since the tweets are being pulled from the API are already in JSON format.

**Displaying Financial Information to the User**

Tweets from a list of pre-defined Twitter accounts that contain financial and investment related information will be displayed to the user in real-time. Creating a database to store the tweets and the information they hold is another primary focus of the project. The application will be able to pull the appropriate data from the database as requested by the user to narrow information based on queries set by the user. The description of each software function is presented below.

## *4.1 Description of Programs*

One program that Twitter Fingers will be responsible for delivering is a program to interact with the Twitter APIs to pull in tweets in real-time. This program will connect to a Twitter Streaming API using our unique API keys and access tokens. This program will stream the tweets and convert them from JSON format so that it can be easily displayed to the user and stored in the database for future access. This program will be written in PHP, and uses the *tmhOAuth* library to connect to the Twitter APIs. A brief description of this library is below:

* **tmhOAuth:** The tmhOAuth library is a PHP library used to connect to many of the various Twitter APIs. There are various methods provided in the library to make streaming requests and pull information from Twitter. This library is preferred because it is written in PHP, and has documentation to help guide development.

There will also be a program that manages how tweets and information can be displayed once they have been shown to the user. Since the tweets will be pulled in real-time and be continuously scrolling, a tweet can be removed from the display in order to make room for another incoming tweet. There may be tweets the user will want to look at again, so they will be able to use queries to filter what information they want displayed. The related tweets they want to display will come from the database and then will be presented to the user. The user will be able to display tweets via relevance, stock symbol, owner credibility, and virtually any other way we see fit. This program will be designed using a DOM Model in order to map out how the incoming tweets will be accessible. The user will also be able to scroll backward on the display and show tweets that were removed from the display to make room for the newest incoming tweets.

There will be a program that manages how our database works. This program will help us to organize, scale, and use information that we store in the database efficiently, and effectively integrate this information into our web application. We will write this program using a combination of programming and scripting languages, specifically Java, JavaScript, and PHP. One of the important features of this program will be its ability to upscale the database. Since the user specifies which accounts they would like to follow, the database should to be able to hold increasing amounts of information that will be coming in rapidly via the live tweets, should there be a large number of accounts being followed. As we look toward the future, it is important that our database has the ability to grow. Our application will only be streaming the tweets of a few Twitter accounts, but we want our application to have the potential to follow thousands of Twitter users, similar to an actual account on Twitter.

We plan to design our web application with using HTML, JavaScript, and CSS. An atheistically pleasing user interface will display the information from our database in an organized way. We will utilize design libraries to have a dynamic display with many features.

There is a possibility that more programs will be introduced as we obtain more feature requirements and feedback from our client during the multiple iterations planned.

## *4.2 Design Constraints*

The design constraints for the software project are as follows:

* **Time**: Most members in our team are not familiar with MongoDB. With the entire project being dependent on the use of this system, time spent learning how to manage the database system will be taken into account.
* **Experience**: There are several tools that our team will be using in this development process. Some of them include as MongoDB, and scripting languages such as JavaScript and PHP. Some of our team members have no experience using any of these tools, so there will be a learning curve when developing with these tools.

*4.3 Software Interface Description*

* **Dashboard**: The web application dashboard will allow the user to easily navigate and use tools to personalize a display feed of tweets to find financial information on Twitter.
* **Account List:** A list of Twitter accounts already contained in this application will inform the user of what will be contained within the scrolling display feed. The user chooses all Twitter accounts within the system, and they can remove or add accounts.
* **Start Button:** This button will activate and resume the scrolling display feed that shows incoming tweets after the stop button has been pressed.
* **Display Feed:** All the gathered information from financial related Twitter accounts will be displayed here in a scrolling feed. Information is gathered and updated continuously in real-time. Information includes twitter screenname, tweet, and hashtags. ex. John123 : “Twitter is so much fun #awesome !”
* **Stop Button:** Stops the scrolling display feed. This is useful if the user would like to view a specific tweet in more detail. The tweets will be continuously added to the feed while the application is stopped and will be resumed when the start button is pressed.

### *4.3.1 Human interface*

The function of this application is simple in nature. The design of the interface should be simple and effective. The range of possible user choices is not substantial and thus the dashboard should not be cluttered. The user has the ability to select the Twitter accounts they want to receive tweets from, start the display, stop the display, specify which particular tweets they want to have displaying, and perform a statistical analysis.

# **5.0 Behavioral Model and Description**

Users should be able to successfully navigate to each function of the application with ease. A dashboard that is organized intuitively and simple menus will allow users to effectively perform the functions this application has been specified to perform. Simplicity, ease of use, and proper functions are a key goal of this project, and will be a metric to determining the success of this product going forward.

## *5.1 Description for software behavior*

This software will contain several events and states, some of which occur automatically through normal use, and some the result of user actions. The main events of this application are centered around starting and stopping the display function, scrolling backward to view previously displayed tweets, organization of tweets using queries, and the analysis of the tweets. The three states for this application include: actively streaming, waiting, and analysis.

### *5.1.1 Events*

There are several events that will occur during the use of this application, including: starting a display feed, modifying display parameters, interaction with the display feed, and the performing of the analysis. The user will also be able to modify display parameters in order to allow for particular tweets/information to be displayed as opposed to displaying all tweets/information. When the display feed begins, the user will be able to stop the real-time display feed at any time, returning the state to waiting until it is started up again. When the display feed has been stopped, the user will be able to scroll backwards and view previously displayed tweets that were removed in order to make room for the newest incoming tweets. This is useful should the user have missed a tweet they may want to go back and look at.

*5.1.2 States*

The initial state of the Web application is **wait** for a user request. The application references a list of pre-defined accounts that the user can add to and subtract from. The user activating the start button activates the **display** state. The application will remain in this state, continuously updating and scrolling to display gathered information until specified to do otherwise by the user. The user can stop the display stream at any time.

# **6.0 Restrictions, Limitations, and Constraints**

The application will be an online application available in a website, not compatible with mobile.

* Users will not be able to access the database or manage it.
* For the database, a NoSQL database must be used.

# **7.0 Version History**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Document** |
| 1.0 | 1/27/2017 | Requirements Version 1 |
| 2.0 | 2/16/2017 | Requirements Version 2 |
| 3.0 | 3/16/2017 | Requirements Version 3 |
| 4.0 | 4/6/2017 | Requirements Version 4 |
| 5.0 | 5/2/2017 | Requirements Version 5 |