Sentiments Analysis with Twitter

TEST PLAN
Team JAZZ MEN

Anagh Goswami 1217426
Meet Pandya 1214306
Jasman Gill 1211554
Jesse Truong 1222722
Jia Xu 1213268

January 11, 2017

Contents

Description of Changes	Author(s)	Date
Created first draft	Jasman Gill	2016-11-02
Edited the information for the new idea	Meet Pandya	2017-01-02

Table 1: Revision History

1 Overview

The purpose of this document is to provide an initial plan for the testing of our project and application. The following gives out a brief outline and an overview of what is discussed in this document:

- A proof of concept is described in section 3

- The different variations of test cases to be tested are described in section 5

- Each test case is categorized based on whether it is for a functional or non-functional requirement in sections 5.1 and 5.2

1.1 Test Case Format

Each test case is formatted as shown below:

Test:	**Test name**
Requirement:	**Requirement Type**
Description:	A description of the test being conducted
Initial State:	Initial State of system before the test begins
Input:	The input to the system that will change the system's state
Output:	The relevant output that is observed
Pass:	The pass criteria describes what is acceptable as a successful test based on the requirements of the system

2 Proof of Concept Testing

3 Significant Risks

To successfully accomplish this project, the risks mentioned below must be taken care of:

1. Reliability of server

2. Effective Web UI Design

3. Ensuring website is Portable (Mobile and Desktop Friendly)

4 Demonstration Plan

For the proof of concept demonstration, we will be making a working prototype consisting of an analysis of sentiments through the key term "McMaster University". We will then use the Bootstrap Framework to develop the front end of the web page in order to display the results to the public.

The prototype will run the sentiment analysis API using the keywords and with the help of the Twitter API, it will sift through tweets using the keywords and generate results. These results will then be displayed in a tabular format for users to view. The score will be formatted in an understandable manner to the users, allowing users to view both positive and negative spectrums of the results (see Figure 1).

5   Testing

5.1   Automated Testing

All unit testing for this project will be automated to save time. Unit tests will be created and tested as the project is developed. These unit tests will be separated by different modules including Python unit testing and Javascript unit testing.

5.2 Functional Requirements Testing

Test:	**Verify Twitter API Parsing (Automated)**
Requirement:	**Functional (Twitter Parsing)**
Description:	Test a keyword to search for through the Twitter API
Initial State:	Home Page of the Web UI, Idle Mode
Input:	A keyword string or query
Output:	JSON data with tweets
Pass:	Verify that each tweet has the keyword at least once to ensure integrity

Test:	**Verify IBM Watson API Parsing (Automated)**
Requirement:	**Functional (Watson API Parsing)**
Description:	Check IBM watson API is able to parse through tweets provided from Twitter API and gives a score
Initial State:	Home Page of the Web UI, Idle Mode
Input:	Twitter data
Output:	Sentimental numeric score
Pass:	Verify score is numeric and verify if the tweet's message corresponds to the score given

Test:	**Web Browser Verification (Manual)**
Requirement:	**Functional (Web Browser)**
Description:	Check web browser's compatibility in order to run the application
Initial State:	Home Page of the Web UI, Idle Mode
Input:	A keyword string or query
Output:	Sentimental Score and a table with list of impactful tweets
Pass:	Ensure the web browser is able to display the results successfully

5.3   Non-Functional Requirements

Test:	**Uniform Theme of Web UI (Manual)**
Requirement:	**Non-Functional**
Description:	Check web UI through all its sections and pages
Initial State:	Home Page of the Web UI, Idle Mode
Input:	Home Page of the Web UI
Output:	Back at Home Page after scanning through all the pages and sections
Pass:	Ensure the look and feel of every area of the Web UI is uniform

Test:	**Usability Test (Automated)**
Requirement:	**Non-Functional (Usability)**
Description:	User is able to personalize their request
Initial State:	Home Page of the Web UI, Idle Mode
Input:	A keyword or a score
Output:	Sentiments Score based on keyword OR Tweets based on score
Pass:	Ensure appropriate data is provided for the two types of filter based on keyword and score

Test:	**Performance (Manual)**
Requirement:	**Non-Functional (Performance/Speed)**
Description:	Time taken between searching and displaying results must be minimal
Initial State:	Home Page of the Web UI, Idle Mode
Input:	A keyword string or query
Output:	Result displayed as sentiments score and impactful tweets displayed in table
Pass:	Ensure the time taken between entering a keyword and seeing results is minimal(few seconds)

Test:	**Precision and Accuracy (Automated)**
Requirement:	**Non-Functional (Reporting)**
Description:	Analyzed data to ensure it is accurate and precise
Initial State:	Home Page of the Web UI, Idle Mode
Input:	A keyword or string query
Output:	Sentiments Score along with the most impactful positive and negative tweets
Pass:	Ensure the dataset has the keyword in each tweet to ensure completeness of the system

Test:	**Fault Tolerance (Automated)**
Requirement:	**Non-Functional (Fault Tolerance)**
Description:	Check for exception thrown when error occurs
Initial State:	Home Page of the Web UI, Idle Mode
Input:	A keyword or query string inputted
Output:	Exception thrown with an error message
Pass:	The error message must be meaningful telling the user exactly what went wrong

6 Timeline

This document is designed in a way to follow the timeline posted below. The timeline is to assist the progression of testing the system and it is an estimated completion time which could or could not be met on time due to running into errors.

Expected Completion Date	Task to be completed
November 20, 2016	Project Idea Changed to Sentiments Analysis
November 25, 2016	Proof of Concept Demonstration completed
January 10, 2016	Document updated with new idea