Software Requirements Specification

for

El Nino Tracker using Social Sensors

Version 1.0 approved

Prepared by Kevin Anthony Martinez

University of the Philippines Los Banos

May 25, 2016

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 2

1.5 References 3

2. Overall Description 4

2.1 Product Perspective 4

2.2 Product Functions 5

2.3 User Classes and Characteristics 5

2.4 Operating Environment 6

2.5 Design and Implementation Constraints 6

2.6 User Documentation 7

2.7 Assumptions and Dependencies 7

3. External Interface Requirements 9

3.1 User Interfaces 9

3.2 Software Interfaces 10

3.3 Communications Interfaces 11

4. System Features 11

4.1 Collect Tweet 11

4.2 Upload Test Cases 12

4.3 Upload Training Data.....…………………………………………………………………………………12

4.4 Train Classifier...………………………………………………………………………………………….13

4.5 Sentiment Analyzer…………………………………………………………………………………….…14

4.6 Classify Data…...…………………………………………………………………………………………14

4.7 Connect to Server Using Socket IO.....…………………………………………………………………15

4.8 Create Map…….........................................................................................................................................15

4.9 Plot Data……….………………………………………………………………………………….16

5. Other Nonfunctional Requirements 16

5.1 Performance Requirements 16

5.2 Safety Requirements 17

5.3 Software Quality Attributes 17

6. Other Requirements 18

Appendix A: Glossary 19

Appendix B: Analysis Models 21

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

The purpose of this document is to give a description of the requirements for the software El Nino Tracker v.1. This document will also explain the interface and constraints of the system and will serve as the user manual for the client and an overview of the development of El Nino Tracker that can be read as reference by both users and developers.

## Document Conventions

## There are no special fonts and highlights specified but there are bolded and italicized phrases for the captions of figures and illustrations that demonstrate some technicalities and interfaces. The words system, application, software, and app will refer to El Nino Tracker system. The word machine will refer to the computer which hosts the system. On the other hand, underlined words are terms that are unfamiliar or given meaning in the context of the system which are defined under Appendix A: Glossary.

## Intended Audience and Reading Suggestions

# This document is intended for four groups of people: the developers who will create and modify the software, the users who wish to know more information of the software, the clients who want an overview of the project and the maintenance team that will need the information of the system in order to update and maintain the project properly.

# The rest of this document includes five other chapters. The second chapter focuses on the overall description of the project, including a list of its features, an assumption of what type of users will use the system, a background of the system’s operating environment and the constraints that the developers may encounter in creating the system.

# The third chapter deals with the interface requirements of the system. This includes characteristics of each interface that the user may encounter while using the system. Furthermore, this includes screenshots that serve as the design template of the interface. Also, this chapter describes the physical characteristics of the interfaces that associates with the hardware used in the system.

# The fourth chapter provides a detailed description of each features and main services that the system will provide for its users. Moreover, this includes the possible limitations and capabilities of a certain feature of the system.

# The fifth chapter specifies certain requirements that is connected with and may affect the overall safety, performance, security and the quality of the system itself.

The sixth chapter, together with the appendices, includes all other requirements not mentioned elsewhere in the previous chapters of the document such as tables, diagrams, database, packaging, and internationalization requirements.

## Product Scope

The Republic of the Philippines is one of the developing countries which are considered "disaster hotspots" due to their settlement in the world's multi-hazard zones and poor economic capacity. Moreover, the emergence of social media platforms in the country made way for a whole different landscape of communication amongst online citizens in times of El Nino. This progress in the emergency response includes the utilization of Twitter for the purpose of communicating information regarding risks and disasters. In this light, several disaster management organizations, as well as the public, took to social networks, especially Twitter for quick response towards rescue and relief operations.

From this perspective, the lack of effort to understand the trend of Twitter topics and conversations leads to a cluttered course of information dissemination, and the attempt to efficiently utilize Twitter becomes futile. Twitter offers a quick-paced pool of content, thus, nuisances that are classified as irrelevant should be filtered out to identify relevant discourses in times of emergency or hazards, especially typhoons. Despite the presence of Twitter tracking and analytic tools, the unavailability of a system that fixates on the detection and prediction of affected areas gives opportunity to maximize the use of Twitter for disaster reduction and crisis management.

Furthermore, development of site-specific sensors can be costly for the government and other disaster management organization. Aside from the development of said sensors, maintenance of these sensors can be challenging. Not only it is expensive, but it will also require additional man power to maintain each sensors.

To address this deficit, this application aims to predict the places affected by El Nino Southern Oscillation (ENSO). Using Twitter Streaming API, the application extracted the tweets coming from the Philippines on dates from March 3, 2016 to April 3, 2016. Due to the limitation of computational resources and the terms available, the tweets extracted will be in English and Filipino languages only. Rather than relying on the data sent by site-specific sensors, the application will use social sensors to predict the areas affected by ENSO. This application is anchored on sentiment analysis to classify the tweets according to an actual occurrence of El Nino. Using the twitter users as sensors and tweets as indicators, the affected areas can be determined.

## References

Aasen, R., Cerauskis, L., Matson, B., Carlisle, E., Jeffers, E., Ritchey, J.Vo, P. (2011). *Software Requirements Specification for SplitPay*. Retrieved December 21, 2013 from http://[www.cise.ufl.edu/class/cen3031fa13/SRS\_Example\_1\_2011.pdf](http://www.cise.ufl.edu/class/cen3031fa13/SRS_Example_1_2011.pdf)

Ambrose, J. (2007). *Software Requirements Specification ForUserTrak -User Tracker, Reporter and Kommander*. Retrieved from: <http://cse.csusb.edu/programs/ba_computer_systems/cse_482/srs/Jambrose-SRS-482-F07.pdf>

Atukorala A.U.B., Priyadasraha G.V.J., Dissanayake C.P., Kumara M.G.C.P. (2007). *Software requirements Specification for MobiTiki*. Retrieved from: <http://lms.uom.lk/sf/shantha/Project-web-sites/2007-08/Grp-04-Mobitiki/Website/documents/SRS.pdf>

Bennett, J., Burgess, M., Huntwork, J.,& Robertson J. (2005).*Alfs Software Requirements Specification*. Retrieved December 28, 2013, from http://[www.linuxfromscratch.org/](http://www.linuxfromscratch.org/)alfs/view/alfs-srs/alfs-srs.html

Chung, L. *Non-functional Requirements*. Department of Computer Science, University of Texas at Dallas.

Retrieved December 21, 2013 from [www.utd.edu/~chung/RE/NFR-18-4-on-1.pdf](http://www.utd.edu/~chung/RE/NFR-18-4-on-1.pdf)

Eaker, F. (2006). *Software Requirements Specification vyasa*. Retrieved December 21, 2013 from <http://vyasa.sourceforge.net/vyasa_software_requirements_specification.pdf>

Geagea, S., Zhang, S., Sahlin, N., Hasibi, F., Hameed, F., Rafiyan, E., & Ekberg, M. (2010). *Software Requirements Specification Amazing Lunch Indicator.* Retrieved December 13, 2013 from[http://www.cse.chalmers.se/~feldt/courses/reqeng/examples/srs\_example\_2010\_group2.pdf](http://www.google.com/url?q=http%3A%2F%2Fwww.cse.chalmers.se%2F~feldt%2Fcourses%2Freqeng%2Fexamples%2Fsrs_example_2010_group2.pdf&sa=D&sntz=1&usg=AFQjCNHvA9IzveWC37Knpqz959Bt72hJaA)

Güçlükol, A., Paçacı, A., Taze, M., Arslanhan, S. *Software Requirement Specification for Web Based Integrated Development Environment*. Retrieved December 21, 2013 from <http://senior.ceng.metu.edu.tr/2013/tintin/documents/SRS.pdf>

Lee, A. How to Write Performance Requirements with Example. Retrieved December 21, 2013 from <http://www.1202performance.com/atricles/how-to-write-performance-requirements-with-example/>

Perrera, S. (2005). *Project ZINX Software Requirements Specification*. Retrieved December 28, 2013, from [http**://**znix.sourceforge.net/vision/SRS.pdf](http://znix.sourceforge.net/vision/SRS.pdf)

# Overall Description

## Product Perspective

The El Nino Tracker is a new, self-contained product intended for use in Windows platform. While the El Nino Tracker is the main focus of the project, there is also a server-side component which will be responsible for database and synchronization services. The scope of the project encompasses on both server- and client-side functionalities, so both aspects are covered in detail within this document. The application relies on sentiment analysis to determine the areas websocket technology to send continuous data from the server. Figure 2.2 shows the interaction of client and server using websocket technology.

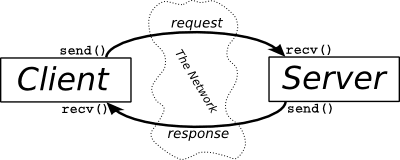
****

Figure 2.1 illustrates the interactions between the server and client applications.

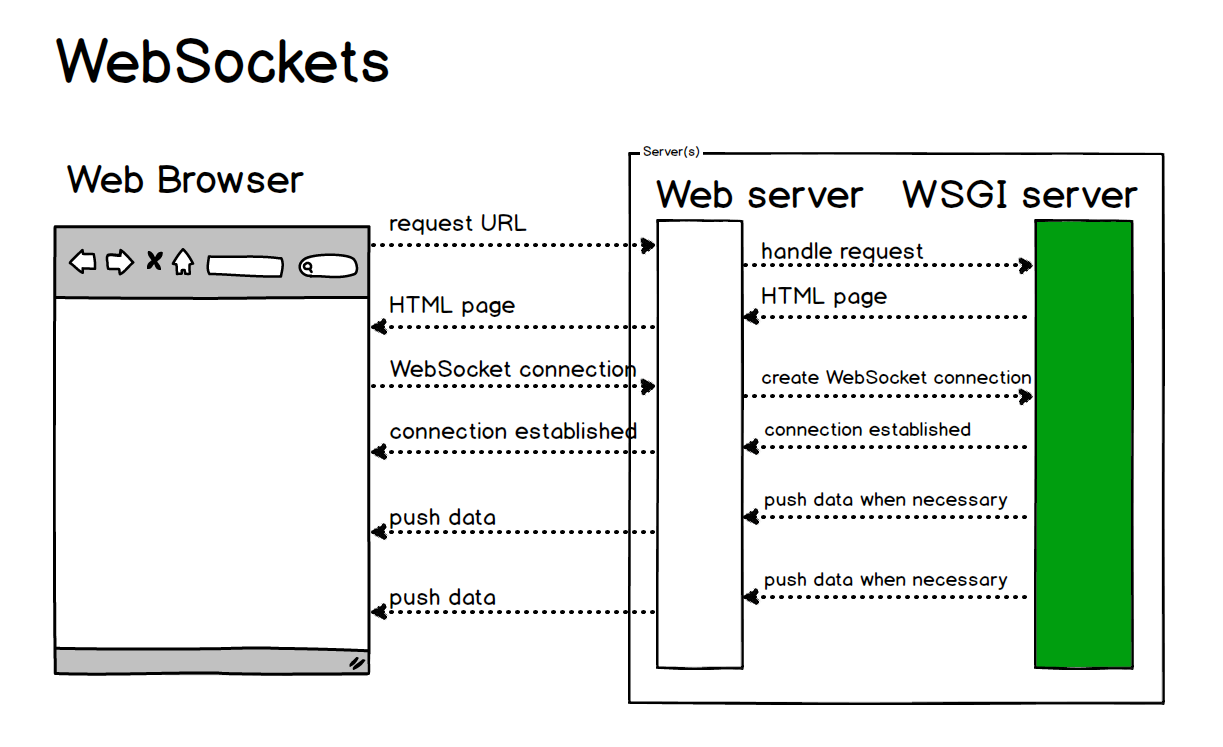


Figure 2.2 shows the interaction of client and server using websocket technology.

## Product Functions

The following list offers a brief outline and description of the main features and functionalities of the El Nino Tracker. The features are split into two major categories: Server-Side features and Client-Side features. Server-Side Features include the essential functionalities of the server whereas Client-Side Features includes functionalities from the client side.

### Server-Side Features

* Collect Tweets
  + Collects the tweets using Twitter Streaming API and saves it to a CSV file.
* Upload Test Cases
  + Uploads the test cases to the database from CSV files. This also creates the database and the tables required for the storage of the test cases.
* Upload Training Data
  + Uploads the training data to the database from TXT files. This also creates the database and the tables required for the storage of the training data.
* Train Classifier
  + Trains the classifier to be used on sentiment analysis.
* Sentiment Analysis
  + Analyzes the test cases based on the training data.
* Classify Data
  + Classifies the test cases into positive or negative classification. This feature uses sentiment analysis to categorize the data.

### Client-Side Features

* Connect to Server via SocketIO
  + Establish a websocket connection to the server.
* Create Map
  + Creates the map of the Philippines where the test cases will be mapped.
* Map Data
  + Maps the coordinates of the test cases

## User Classes and Characteristics

The El Nino Tracker is meant to provide an a map that displays the predicted areas affected by ENSO. Consequently, the system has a very little or no learning curve at all, for there will be a self-explanatory user interface, plus an additional Contact page which contains the FAQ in the online repository.

### Users

**Key Functions:**

* Collect tweets
* Upload tweets to database
* Upload training data to database
* Train a classifier
* Classify data
* View classified data

## Operating Environment

The software El Nino Tracker is accessible in Windows 32-bit and 64-bit platform as long as Python 2.7 is installed. The operating system should also have a web browser that supports JavaScript, CSS and HTML and must be able to host the application locally for it to function effectively.

The application uses MySQL database to store the training and test data. It is required that the operating system must have a software that hosts MySQL, preferably WAMP or XAMP. External libraries from Python were used for the development of the application. This includes NLTK, SciPy, Pip, and VirtualEnv to name the few. The software dependencies of this application will be discussed in details in section 2.7.

## Design and Implementation Constraints

Python 2.7 was used in the development of this software along with the external libraries available for the language mentioned (to be elaborated in section 2.7). For the software to run efficiently, it is advisable to have at least an Intel i5 processor or an AMD A10 processor to be able to perform heavy computation for classifying the data and mapping the data in the map in a form of heat map. The software will also be constrained by the amount of memory. The machine should have at least 4 gigabytes (GB) of memory to handle the load of real-time sending of data and plotting of the data to the map. Internet connection is also a constraint for the software. Since the software access the external Javascript files over the Internet, it is vital that the connection is available for the application to function.

## User Documentation

The system will consist of features that will help users gain more information about the system. It will also be helpful for the users in terms of accessing the system.

A “Read Me” text file will be included in the deployment package. This file contains information on how to run the system as well as the list of the external libraries that the system uses. A link to the online repository web page will also be included in the deployment package. The web page will contain a tutorial on how to run the program as well as additional information of the system.

Should there be any issues; the online repository web page will contain contact information of the developer. The problem or question will be sent to the developer which will then be answered as immediately as possible. Also, the online repository web page will include Frequently Asked Questions (FAQ). This will contain the questions and answers that most of the users will like to know. This is a great help for the developer as well because answering similar questions repetitively will be minimized.

## Assumptions and Dependencies

### Hardware Dependencies

Features mentioned are hardware dependent. As mentioned in section 2.5, the machine should have an Intel i5 or AMD A10 processor and have at least 4GB of memory to handle heavy workload of computation and graphing.

### Browser Dependencies

As it was mentioned earlier El Nino tracker is an Internet dependent system and relies on web browsers to access the contents, thus it is safe to assume that users will have their own preferred web browsers to use in accessing the said system. The system is a web browser based system, thus the system must work properly, even only, in the recent versions of popular web browsers available. Moreover, the browsers should be able to support Javascript, CSS, and HTML.

### Software Dependencies

El Nino Tracker was developed with Python 2.7. As such, external libraries were used in the development. These libraries are:

* Flask 0.10.1
  + - A library used to create a localhost server in Python.
* Flask Socket IO
  + - A plugin in Flask that implements websocket.
* Itsdangerous 0.24
  + - Error detection used by Flask Socket IO.
* Jinja 2.7.3
  + - One of the frameworks where Flask is based from.
* MarkupSafe 0.23
  + - Error detection used by Flask Socket IO
* Python-engineio
  + - A plugin for Flask Socket IO that contains the functions used to execute websocket
* Python-socketio
  + - A plugin for Flask Socket IO that contains the fundamental functions for web socket
* Six 1.9.0
* Werkzueg 0.10.4
  + - One of the frameworks where Flask is based from.
* Eventlet 0.19.0
  + - A library used by Flask Socket IO to establish a connection for websocket.
* NLTK 3.2.1
  + - A library that contains Natural Language tool kits. This library also includes a built in sentiment analyser wrapper.
* Scipy 0.17.1
  + - Contains the sentiment analyser that the system uses.
* MySQLdb
  + - A library used for MySQL query transactions.
* Numpy+INTEL
  + - A library used to Scipy for computing. This contains functions used for computations
* Pip
  + - The PyPA recommended tool for installing Python packages and libraries.
* VirtualEnv
  + - A library that creates a virtual environment where the system is run.

Figure 2.4 shows the dependencies. Furthermore, the machine should have WAMP or XAMP installed in order to host MySQL database.

Test Cases

Uploader

Test Cases

Main Program

Sentiment Analyzer

Trainer

SciPy

NLTK

Training Data

Training Data Uploader

Flask

Flask Socket IO

Figure 2.3 Dependency Diagram of the El Nino Tracker

# External Interface Requirements

## User Interfaces

The User Interface (UI) for the system consists only of command prompt, and the homepage for displaying the data. As shown in Figure 3.1, users can use the command prompt to run the system by inputting text commands.

After running the program, users can access the web using 127.0.0.1:5000 to view the data being plotted in the map. Upon accessing the site, the homepage will be displayed. The homepage contains the map of the Philippines and the data plotted as a heat map as shown in Figure 3.2.

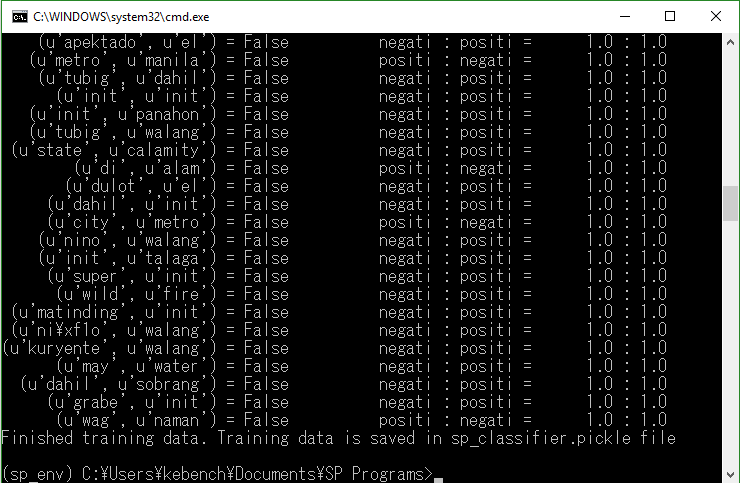


Figure 3.1 User Interface for running the software

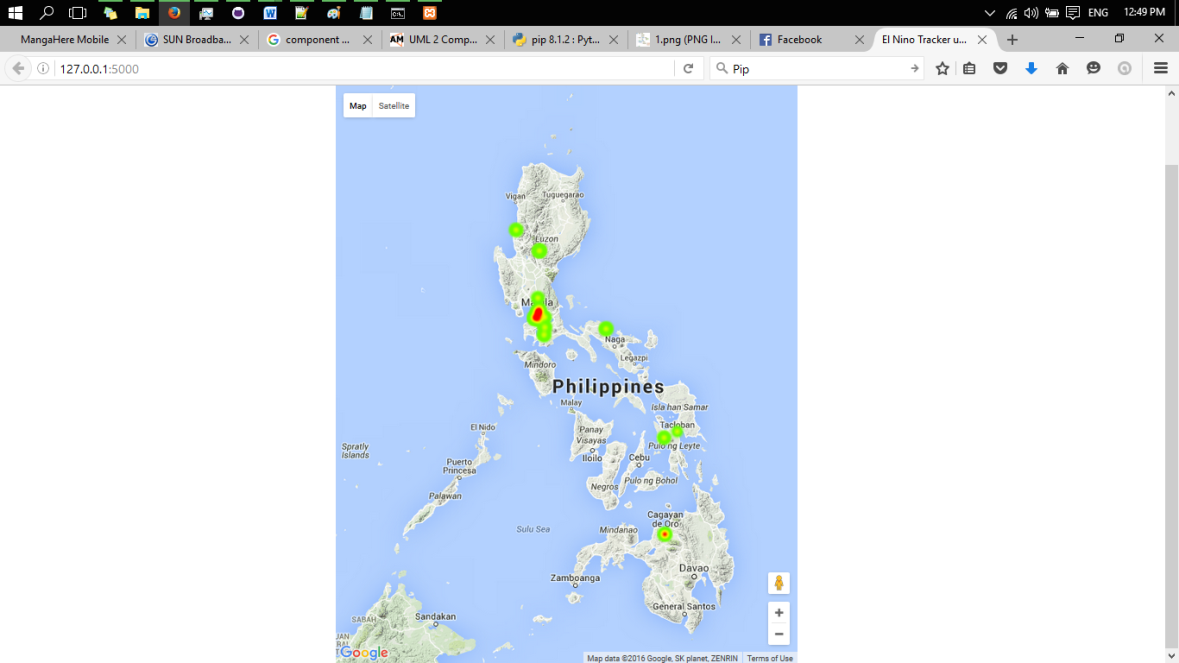


Figure 3.2 Graphical User Interface for displaying the data

## Software Interfaces

### Python Libraries

El Nino Tracker was developed using Python 2.7 with the help of external Python libraries. In order to use the external libraries, the system needs to install Pip. Installing Pip will help in downloading and installing the external libraries. Another important package is Virtualenv. The software should run in a virtual environment to ensure that it is isolated and does not affect any program, libraries or packages.

The application uses sentiment analysis in classifying the data. In order to do so, it needs external libraries. One important library that the software uses is NLTK. The aforementioned library extracts the features needed by the classifier for training the data. The library also creates the classifier for the sentiment analyzer. SciPy is also used in sentiment analysis. It contains the classifier in which the NLTK creates. SciPy library is also in charge of the computation in classifying the data. After the analysis, the classification of the data is relayed back to the software.

In order to send and display the data, the application uses websocket technology. This is implemented through the help of Flask and Flask Socket IO. Flask deploys a localhost environment where it creates a server from the local machine. Flask Socket IO creates the websocket protocol and connects it to the client.

### Other Software

The application can use WAMP, XAMP or other software that deploys MySQL database on a local machine. Once the MySQL database is deployed, the application can store the test and training data for the sentiment analysis.

## Communications Interfaces

The application implements websocket technology in sending and displaying the data. A WebSocket connection allows full-duplex communication between a client and server so that either side can push data to the other through an established connection. The application used this technology so that the server can send the data even when the client is not requesting it. Using this technology, the application can simulate real-time sending of data.

In order to implement the technology mentioned, the application needs the Flask and Flask Socket IO for the server side and socketIO.js for the client side. Once the requirements are satisfied, application can now send the data in a real-time like manner.

# System Features

## Collect Tweet

### Description and Priority

Collects tweets using Twitter Streaming API and saves it to a CSV file.

Priority: Medium

### Stimulus/Response Sequences

Running the tweet\_collect.py script will collect the tweets from twitter stream for 15 minutes. After collecting the tweets, it will save the tweets in a CSV file and prompt the user.

### Functional Requirements

**FR-001:** Twitter Streaming API. The user must have the necessary information such as API key and API secret required to extract the tweets from twitter. Without these, the application will prompt the user to enter the API key and secret.

## Upload Test Cases

### Description and Priority

Imports the test cases to the database from the CSV. This creates the database or the table if any of the mentioned is not present.

Priority: Low

**4.1.2 Stimulus/Response Sequences**

Running the csv\_read.py in the command prompt will import the tweets to the database. Imported tweets will be saved into test\_cases table in sp\_data database.

**4.1.3 Functional Requirements**

**FR-001:** Database System. The database system should be running before running the script. Unable to do so will result to an error, “Error in Connecting to MySQL Database”.

**FR-002:** CSVfiles**.** CSV containing the tweet should also be present in the same folder as the script. If the CSV files are not found, then it will result to an error, “Files not found.” Invalid inputs will result to wrong database data.

## Upload Training Data

### Description and Priority

Imports the training data from text files in the corpora folder into database.

Priority: Low

### Stimulus/Response Sequences

Running the import\_training\_data.py in the command prompt will import the training data to the database. Imported tweets will be saved into test\_data table in sp\_data database.

### Functional Requirements

**FR-001:** Database System. The database system should be running before running the script. Unable to do so will result to an error, “Error in Connecting to MySQL Database”.

**FR-002:** CSVfiles**.** CSV containing the tweet should also be present in the same folder as the script. If the CSV files are not found, then it will result to an error, “Files not found.” Invalid inputs will result to wrong database data.

## Train Classifier

### Description and Priority

Extracts the training data from the database and use it to train the classifier. Trained classifier information will be stored in a .pkl file for the sentiment analyzer to use in the execution of the main program.

Priority: High

### Stimulus/Response Sequences

Running the trainer.py script will train the classifier for the sentiment analysis. Training the classifier might take a while depending on the training data supplied to the classifier. Once the training is complete, information about the trained classifier will be saved in the sp\_classifier.pkl file.

### Functional Requirements

**FR-001:** Database System. The database system should be running before running the script. Unable to do so will result to an error, “Error in Connecting to MySQL Database”.

**FR-002:** sentiment\_analyzer.py**.** Sentiment analyzer class should be imported into the script otherwise it will display an error, “Unknown Module”

## Sentiment Analyzer

### Description and Priority

Classifies the test data depending on the information of the classifier. This includes the type of classifier used and the features of the training data. This class also contains the functions needed for training and classifying the data.

Priority: High

### Stimulus/Response Sequences

Importing sentiment\_analyzer.py to trainer.py will train the classifier based on the training data saved in the database.

Importing the sentiment\_analyzer.py to main.py, the main function of the software, will classify the data based on the information of the trained classifier.

### Functional Requirements

**FR-001:** NLTK Python Library. This feature heavily relies on the NLTK library for Python. Without it, it will not be able to train or classify the data.

**FR-002:** SciPy**.** SciPy contains the classifier for the sentiment analyzer and is responsible for the computation in classifying the data.

## Classify Data

### Description and Priority

Classifies the test data depending on the information of the classifier. This feature uses the functions found in Sentiment Analyzer class. Once the classification is done and the result is positive, it will send the data to the client.

Priority: High

### Stimulus/Response Sequences

Running main.py script will classify the tweets from the test\_cases table. After classification and if the result is positive it will send the data to the server.

### Functional Requirements

**FR-001:** Flask. Flask creates a localhost server in order to simulate the real-time sending of data. The classified tweet will not be sent if Flask is not imported

**FR-002:** Flask Socket IO**.** Sets up websocket so that server can send the data without closing the connection.

**FR-003:** sentiment\_analyzer.py**.** Contains the functions needed to classify the tweets. Failure to import would result to data not being classified.

## Connect to Server using SocketIO

### Description and Priority

Connects the web browser to the server and establishes a websocket connection. Displays the homepage if connection is successful.

Priority: Medium

### Stimulus/Response Sequences

Accessing 127.0.0.1:5000 using a web browser will automatically establish a websocket connection. This will display the homepage as well as the map of the Philippines.

### Functional Requirements

**FR-001:** SocketIO.js. SocketIO.js should be present in order to establish a websocket connection

**FR-002:**Google Maps API**.** Google Maps API sets and displays the map.

## Create Map

### Description and Priority

Creates the map of the Philippines using Google Map API.

Priority: Medium

### Stimulus/Response Sequences

Accessing 127.0.0.1:5000 using a web browser will automatically establish a websocket connection. This will display the homepage as well as the map of the Philippines.

### Functional Requirement

### FR-001:Google Maps API. Google Maps API sets and displays the map.

## Plot Data

### Description and Priority

Plots the data to the map in a form of a heat map using Google Map API.

Priority: Medium

### Stimulus/Response Sequences

Accessing 127.0.0.1:5000 using a web browser will automatically establish a websocket connection. This will display the homepage as well as the map of the Philippines. Once the data is received, it will plot the data to the map.

### Functional Requirement

### FR-001:Google Maps API. Google Maps API sets and displays the map. This also plots the data to the map.

# Other Nonfunctional Requirements

## Performance Requirements

**5.1.1 Response Time**

The expected response time by the system is ideally 2-3 seconds from classifying the tweet, to sending the data to the client, and plotting the data to the map. This is specifically necessary when sending the data to the client. On the other hand, the time for the browser to render the received information may be dependent to the speed of the machine, but the server side is expected to have accomplished sending necessary information without interrupting the user’s flow of thought.

**5.1.2 Workload**

A large number of data are assumed to be processed by the system, but it is expected to handle a large number of data. Ideally, the system should be able to handle up to 50 data per second from both the classification and the plotting of the data.

## Safety Requirements

The system must ensure that the information stored in the database is always correct and updated to avoid misleading the users. Erroneous data in the system may lead to possible losses on the side of the user as well as the integrity of the system to deliver the expected output.

Inserting data into the database is a very critical step since search queries will be dependent on the values stored in the database. If a value for a certain field got misspelled, it may affect the classification of the data thus, affecting the precision and accuracy of the system.

## Software Quality Attributes

**5.3.1 Availability**

The system is expected to be accessible for use 24 hours a day and 365 days a year to anyone who wishes to use the system.

**5.3.2 Correctness**

The features or services offered by the system must do what it is individually required to do base on the requirement specifications stated in this document, under any circumstances and inputs, accurately.

**5.3.4 Reliability**

The system is expected to be reliable at all times. It is expected to achieve 99% uptime.

**5.3.5 Maintainability**

The system is expected to be easily maintainable. It is also expected that the system will be maintained at least once every four months to ensure that the content, especially of the libraries are still up to date. The maintenance releases are aimed to be offered to end-users prior to the start of a semester.

**5.3.6 Reusability**

The data and records stored and utilized by the system can be reused specifically when needed by other functions as well as those that will be added in the future.

**5.3.7 Testability**

The system is expected to run and response excellently to all possible test cases that will be used for testing the application.

# Other Requirements

**6.1 Database Requirement**

The database of the system will be using MySQL. The following are some information stored in the database:

**Test\_cases:** ID, Text, Coordinates, Username, Country Code, Language

**Test\_data:** ID, Text, Category

**6.2 Packaging Requirement**

The system shall be packaged along with source code and all documentation and shall be available for electronic transfer as a single compressed file.

**6.3 Internationalization**

The system supports English language as of the moment.

**6.4 Licensing Requirement**

The El Nino Tracker will be licensed under GNU General Public License.

**6.5 Legal, Copyright and Other Notices**

All propriety trademarks that will be used in this document are copyright of their respective owners.

Appendix A: Glossary

**CSS** refers to Cascading Style Sheets.

**Database** refers to an application that manages data and allows fast storage and retrieval of that data.

**El Nino Southern Oscillation** is a naturally occurring phenomenon that involves fluctuating ocean temperatures in the equatorial Pacific.

**FAQ** stands for Frequently Asked Questions.

**FR** stands for Functional Requirement.

**Gigabytes** used for measuring storage capacity.1GB **=** 1024 Megabytes.

**Graphical User Interface** is a type of interface that allows users to interact with electronic devices through graphical icons and visual indicators.

**Hardware** is a comprehensive term for all of the physical parts of a computer, as distinguished from the data it contains or operates on, and the software that provides instructions for the hardware to accomplish tasks.

**HeatMap** is a graphical representation of data where the individual values contained in a matrix are represented as colors. Map used to represent intensity.

**HTML** refers to Hypertext Markup Language.

**Javascript** is a client-side scripting language used to create dynamic interactions of a website.

**MySQL** is one of the most used database.

**Operating System** is a collection of software that manages computer hardware resources and provides common services for computer programs.

**Platform** is an underlying computer system on which application programs can run.

**Repository** isa central location in which data is stored and managed.

**Response Time** is the time between a user request and a response to the request from the system.

**Sentiment Analysis** refers to the use of natural language processing, text **analysis** and computational linguistics to identify and extract subjective information in source materials.

**Sentiment Analyzer** is a tool used for sentiment analysis.

**Software** means [computer](http://www.webopedia.com/TERM/C/computer.html) [instructions](http://www.webopedia.com/TERM/I/instruction.html) or [data](http://www.webopedia.com/TERM/D/data.html). Anything that can be [stored](http://www.webopedia.com/TERM/S/store.html) electronically is software.

**Tweets** isa posting on the social media website Twitter.

**Twitter** is a micro-blogging site. One of the popular social media

**Twitter Streaming API** is an Application Program Interface used to extract the tweets from twitter.

**User** is someone who borrows books from ICS Library or someone who accesses the online system.

**User Interface** is the means by which the user and a computer system interact, in particular the use of input devices and software.

**Web Browser** is a software application for retrieving, presenting and traversing information resources on the World Wide Web, including [Web pages](http://www.webopedia.com/TERM/W/web_page.html), images, video and other files.

**Websocket** is a protocol providing full-duplex communication channels over a single TCP connection.

**Workload** is the amount of processing that the computer has been given to do at a given time.

Appendix B: Analysis Models

Database

Upload Training Data

Main Program Classifier

Trainer

Upload Test Cases

Training Data CSV Files

Sentiment Analyzer

Test Cases CSV Files

Plot Data to the Map

Collect Tweets

Classifier Info .pkl file

Figure A-1 Data Flow Diagram of El Nino Tracker

|  |
| --- |
| Test Cases |
| ID  Text  Coordinates  Created\_at  Screen\_name  Country\_code  Lang |

|  |
| --- |
| Test Data |
| ID  Text  Category |

Figure A-2 Database Tables Used for El Nino Tracker