

TWASTER FINAL PROJECT REPORT

SWE 573 -Software Development Practice

13.05.2018

Ugurcan Akpulat

Table of Contents

1. PROJECT SUMMARY	3
2. INVESTIGATION.....	4
3. EVENT DETECTION.....	5
3.1. SUPPORT VECTOR MACHINE TO INCREASE PRECISION	6
Why SVM is chosen	6
4. LOCATION DETECTION	9
5. PROJECT PROGRESSION.....	9
5.1. TWITTER API AND GIT RESEARCH	9
About Using Git.....	9
About Twitter Api	10
5.2. REQUIREMENT PHASE	10
5.3. PLANNING PHASE	11
5.3. Mock-ups	12
5.4. Use Case Scenarios.....	13
A daughter far away	13
Crime and hope of disasters	14
5.5. Design Phase	14
Class Diagram	14
Database Design	15
5.5.1 Sequence Diagrams.....	16
Earthquake Detection	16
User Registration	16
User Login	17
Password Reset.....	17
6. DEPLOYMENT	18
7. CONCLUSION & FUTURE WORK.....	19
8. REFERENCES	19

1. PROJECT SUMMARY

When an earthquake occurs it generates seismic waves which radiate away from the rupture point like waves in a pond, but also travelling downwards through the earth. Scientists can detect and measure these vibrational waves at great distances.

Mexico City has an EEW system that warns of strong shaking from large earthquakes that occur off of the country's coast. The system consists of a series of sensors located along the coast that detect shaking from a large earthquake and rapidly determine the location and magnitude. Since Mexico City is located several hundred miles from the main plate boundary they can receive up to a minute or more of warning of the impending shaking for subduction zone earthquakes, but warning times are shorter for earthquakes that occur closer to the city. This system has been in operation since 1991.

Japan currently has the most sophisticated early warning systems in the world. The warnings were initially developed for use in slowing and stopping high-speed trains prior to strong shaking. The success of that program in addition to the devastating effects of the 1995 Kobe earthquake paved the way for building a nationwide early warning system. Japan has built a dense network of seismic instruments to rapidly detect earthquakes. They have been issuing public warnings since 2007.^[1]

Besides traditional methods, some scientists searched for different ways to detect an earthquake as early as possible. One of them is detecting earthquakes from social media, especially using Twitter data.

Twitter, a popular microblogging service, has received much attention recently. An important characteristic of Twitter is its real-time nature. For example, when an earthquake occurs, people make many Twitter posts (tweets) related to the earthquake, which enables detection of earthquake occurrence promptly, simply by observing the tweets.

In this project, earthquakes will be detected solely using Twitter information. In fact, after deployment of the project at April 21, 2018 more than thirty earthquakes were detected successfully around the world. Most successful detection happened in Oakland within just 12 seconds. Detection can vary according to location of where earthquake happens. If it's in rural areas the time is up to 60 seconds. But if the epicentre is close to big cities where Twitter usage is higher it is possible to detect earthquakes between 12 to 29 seconds.

2. INVESTIGATION

Earthquake alert dissemination is a straightforward use of Twitter. Broadcasting earthquake alerts via Twitter has several advantages. It provides an additional method to reach large segments of the population who are becoming more reliant on social media as a means of communication and information gathering. Twitter is fast; Twitter messages are generally available to all followers within seconds of being submitted. Users leverage Twitter's delivery infrastructure such that once a message delivery is set up there is no additional overhead for the user to reach more people. In several countries, Twitter also allows users to receive tweets via simple message service (SMS) text messages on their mobile phones at no cost to the sender.[2]

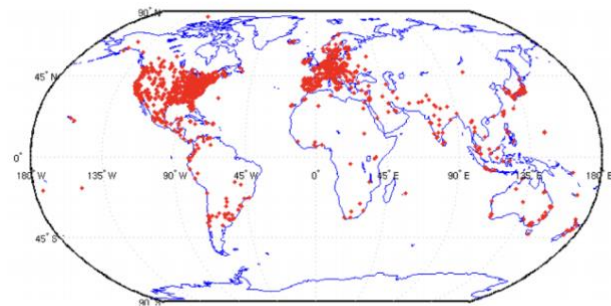


Fig. 1. Twitter user map.

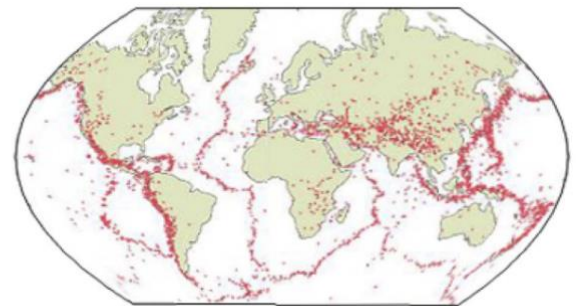


Fig. 2. Earthquake map.

Fig. 1 portrays a map of Twitter users worldwide; Fig. 2 depicts a map of earthquake occurrences worldwide. It is apparent that the only intersection of the two maps, those regions with many earthquakes and large Twitter users, is Japan. Other regions such as Indonesia, Turkey, Iran, Italy, and Pacific coastal US cities such as Los Angeles and San Francisco intersect. [3]

3. EVENT DETECTION

An event is an arbitrary classification of a space-time region. An event might have actively participating agents, passive factors, products, and a location in space/time. We target events such as earthquakes, typhoons, and traffic jams, which are readily apparent upon examination of tweets. These events have several properties.

1. They are of large scale (many users experience the event).
2. They particularly influence the daily life of many people (for that reason, people are induced to tweet about it).
3. They have both spatial and temporal regions (so that real-time location estimation is possible). Such events include social events such as large parties, sports events, exhibitions, accidents, and political campaigns. They also include natural events such as storms, heavy rains, tornadoes, typhoons/hurricanes/cyclones, and earthquakes. We designate an event we would like to detect using Twitter as a target event. In this section, we explain how to detect a target event using Twitter. First, we crawl tweets including keywords related to a target event. From them, we extract tweets that certainly refer to a target event using devices that have been trained with machine learning. Second, we detect a target event and estimate the location from those tweets by treating Twitter users as “social sensors.”

In our case the keyword for detecting earthquake is simply “earthquake”. Couple of combinations of different keywords has been investigated (ex. “shaking”, “moving”, “jold” and “shock”) but they are not seemed not relevant enough. People are using these keywords for their daily actions so the false information ratio drastically increase when using such keywords.

But even the chosen keyword is true there is not a way to understand solely if the earthquake tweet made by real people and another problem was people are keep talking after an earthquake occurrence and those tweets are also considered sensor information without classifying them as actual information. For these purpose the application need to be more precise. To understand the source of tweet is an actual human, a classification technique in machine learning has chosen.

3.1. SUPPORT VECTOR MACHINE TO INCREASE PRECISION

“Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. It is mostly used in classification problems.

Why SVM is chosen

- Effective in high dimensional spaces.
- Still effective in cases where number of dimensions is greater than the number of samples.

- Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.

First I was in need of data to classify tweets. So I managed to download tweets back in 2016 about an earthquake in CA. We took tweeter users as sensors but like many mechanical or electrical sensors in the world it's possible that they publish false information. So here is what I do.

	eq	httpsOrAtSymbol	length	position	text
1	0	1	12	0	Earthquake M6.8: Ferndale, California via @quakefeed - To close to home ...
2	0	1	7	0	#Earthquake M6.8: Ferndale, California via @quakefeed #quake
3	0	1	8	1	6.8 #earthquake if coast of California#Ferndale #breakingnews pic.twitter.com/2DCixcfe0q
4	0	1	20	1	#BREAKING #EARTHQUAKE M6.8 - 167km W of #Ferndale , #California 1449 GMT EVENT possible tsunami info http:// earthquake.usgs.gov/earthquake
5	0	0	21	19	M6.8 - 167km W of Ferndale, California 08/12/16 09:49:45 EST [UD: 08/12/16 09:54:11 EST] Depth: 10km (6.21mi) #hmrdr #Quake #Earthquake eq_
6	0	1	3	1	#California #Earthquake pic.twitter.com/b8LW0EQJzz
7	0	1	6	0	Earthquake M6.8: Ferndale, California via @quakefeed
8	0	1	18	0	#EARTHQUAKE New Alert: Earthquake - 6.8 - 167km W of Ferndale, California, Severity: WARNING http:// snc.pdc.org/PRODUCTION/701 35508-c958-
9	1	0	7	4	Anybody else feel that earthquake just now?
10	0	1	6	0	#Earthquake M6.8: Ferndale, California via @quakefeed
11	0	1	6	0	Earthquake M6.8: Ferndale, California via @quakefeed
12	1	0	2	0	Earthquake ahhh!
13	0	1	12	1	Mwp6.5 #earthquake Off Coast of Northern California 2016.12.08-14:49:48UTC (40.5,-126.3,10km) http:// early-est.alomax.net pic.twitter.com/n5jwh
14	0	1	6	0	Earthquake M6.8: Ferndale, California via @quakefeed
15	0	1	6	0	Earthquake M6.8: Ferndale, California via @quakefeed
16	0	0	19	18	Bill the drunk bought a diesel truck and then got in a fight before waking up to an earthquake
17	0	1	12	9	Cuaca Ekstrim Banget Hari Ini Why ? #haarp #climatechange #earthquake https://www. Instagram.com/p/BNwqKngJ5cv/
18	1	0	6	2	When the earthquake wakes you up
19	0	1	6	0	Earthquake M6.8: Ferndale, California via @quakefeed
20	0	1	7	0	Earthquake M6.8: Ferndale, California via @quakefeed @CNNEE
21	0	1	7	0	#Earthquake M6.8: #Ferndale , #California via @quakefeed
22	0	1	6	0	Earthquake M6.8: Ferndale, California via @quakefeed
23	1	0	7	4	Nothing quite like the earthquake alarm clock
24	1	0	8	5	y'all i actually felt that earthquake this time
25	1	0	5	4	There was just an earthquake
26	0	1	7	1	#sismo #Earthquake M6.8: Ferndale, California via @quakefeed
27	0	1	7	0	#Earthquake M6.8: #Ferndale , #California via @quakefeed
28	0	1	6	0	Earthquake M6.8: Ferndale, California via @quakefeed
29	0	1	5	0	Earthquake M6.8: Ferndale. California pic.twitter.com/esFbKoDGoX
30	0	1			

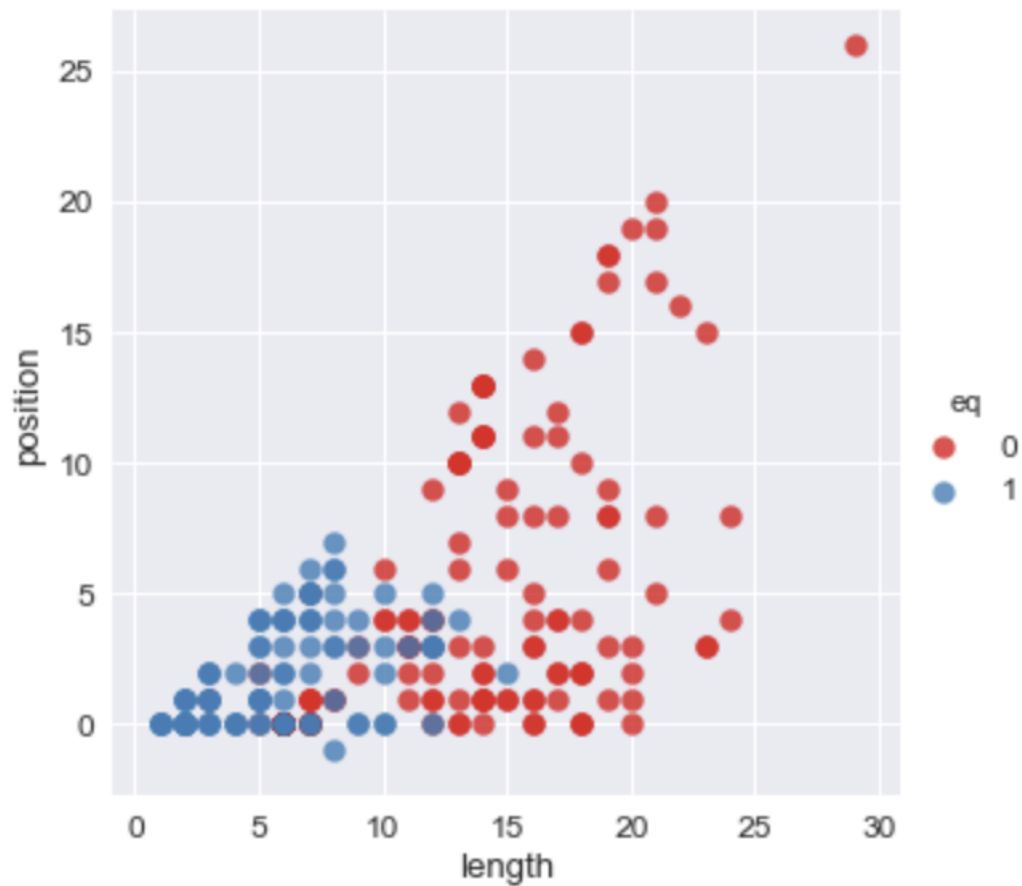
Ss 1 Earthquake related tweets for svm

Three class is chosen to classify tweets for detection.

httpsOrAtSymbol: If a tweet includes a mention to somebody it is very unlikely be an actual earthquake tweet. I have examined more than one thousand tweet by eye and non of human made tweets mentioned somebody in such an hurry about earthquake. So we directly determines mentions as fake information thus marked with 0. Also it's the same for http or https. If a tweet include those keys inside it is also very unlikely an actual event.

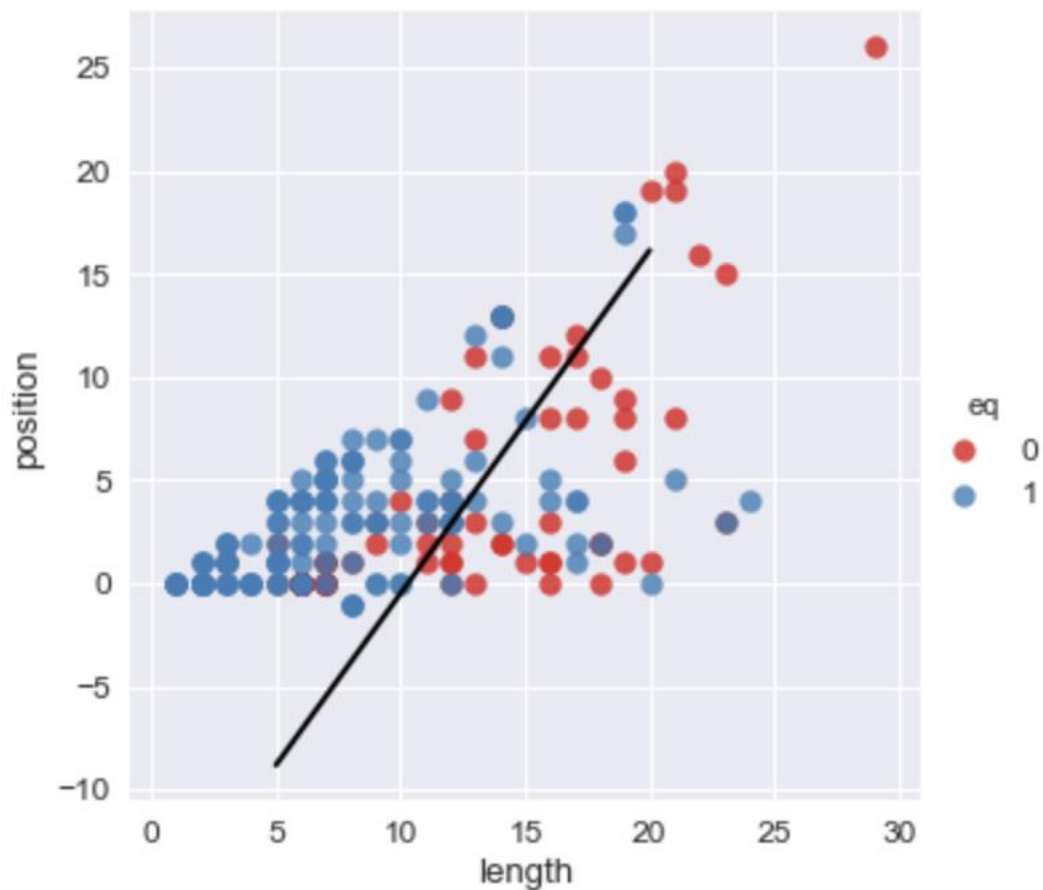
Length: It's observed that tweets coming after an earthquake occurrence are usually short. So these class gets the length of the tweets as an information.

Position: It's also observed that the chosen keyword for event is used earlier in sentence.



Graph 1 length and position classification for earthquake using svm

As you can see the data are separated for some degree but there is some places with both false and true information. So svm draws a line between there occurances and decides when a new info belongs to which place according to where it appears in given graph.



Graph 2 line of svm

The dots above the line considered as earthquakes and which are below are considered false sensor data. So we can't exactly decide with just one tweets as you can see above. But consider a lots of tweets coming from the same place. So chances of an earthquake occurrence drastically improves.

Each tweet has its own post time. When a target event occurs, how do the sensors detect the event? We describe the temporal model of event detection.

According to data I fetched from twitter it seems the tweets about earthquake demonstrate exponential distribution.

$$p_{occur}(t) = 1 - p_f^{n_0(1-e^{-\lambda(t+1)})/(1-e^{-\lambda})}.$$

We can calculate the probability of event occurrence if we set lambda to 0.34 and pf to 0.35.

4. LOCATION DETECTION

Some tweets include geolocation information inside tweet itself. But it is only %1 of all tweets. So, using solely geolocation of tweets was not enough to detect the location of any earthquakes.

There is also some other information about in tweet object itself. Which is PLACES. Some people choose their places when sending a tweet. Twitter includes that chosen places coordinates into account and sent it with the tweet object. In Twitter API you can harness this power also but it was not enough. Because again only %4 of tweets included the places.

Another solution was detecting a using location which user specifies by typing his or her location. Those location information used as string for searching the place in Google Maps API. The results were promising but was a little tricky because let's say there was some tweets coming from Antarctica while earthquake is happening in CA.

I have decided to use all of those information combined and create an algorithm which detects earthquakes without a problem. These algorithm listens tweets coming from SVM module and use them as it's source. It basically creates a min tweet count threshold which changes in time. Simply explained it is actually calculating density of tweets in given time range (which is 30 seconds after first successful tweet returned from SVM module). If density is higher than threshold that means something unusual is happening at the location where those tweets are coming from. The tweets may come from different cities or even different countries because of arbitrary location information but algorithm gets the most used location and successfully sent notification to users which are following the city from our Twassterapp.com.

5. PROJECT PROGRESSION

5.1. TWITTER API AND GIT RESEARCH

About Using Git

Fortunately, I have used git version control system before. It was not hard for me to create a git hub repository. But my git knowledge was really limited. And I have used it just mainly with user interface. Now I decided to go for command line usage which was surprisingly easy and efficient.

First of all I have found an amazing Youtube tutorial which shows git commands in practise. "Learn git in 20 minutes" and the other spectacular Youtube tutorial which explains the git workflow in detail. "GIT Workflow - Georgia Tech - Software Development Process"

After watching these two I was able to perform git commands easily and of course understand what I do in the process.

About Twitter Api

First I have searched information about what is a rest API. When I got the logic of it I started to read twitter API docs. It seems you can do anything with twitter API. Like sending tweets. Or searching tweet histories for getting large amount of information. Possibilities seems limitless. Before start typing any code I have decided to use the API and see if it is working or not? Using //apps.twitter.com I have created my tokens. And created a post request using post-man application to send a tweet.

5.2. REQUIREMENT PHASE

At this point I have decided twasterapp.com requirements, which are changed drastically in time. Now here are some basic requirements.

User requirements:

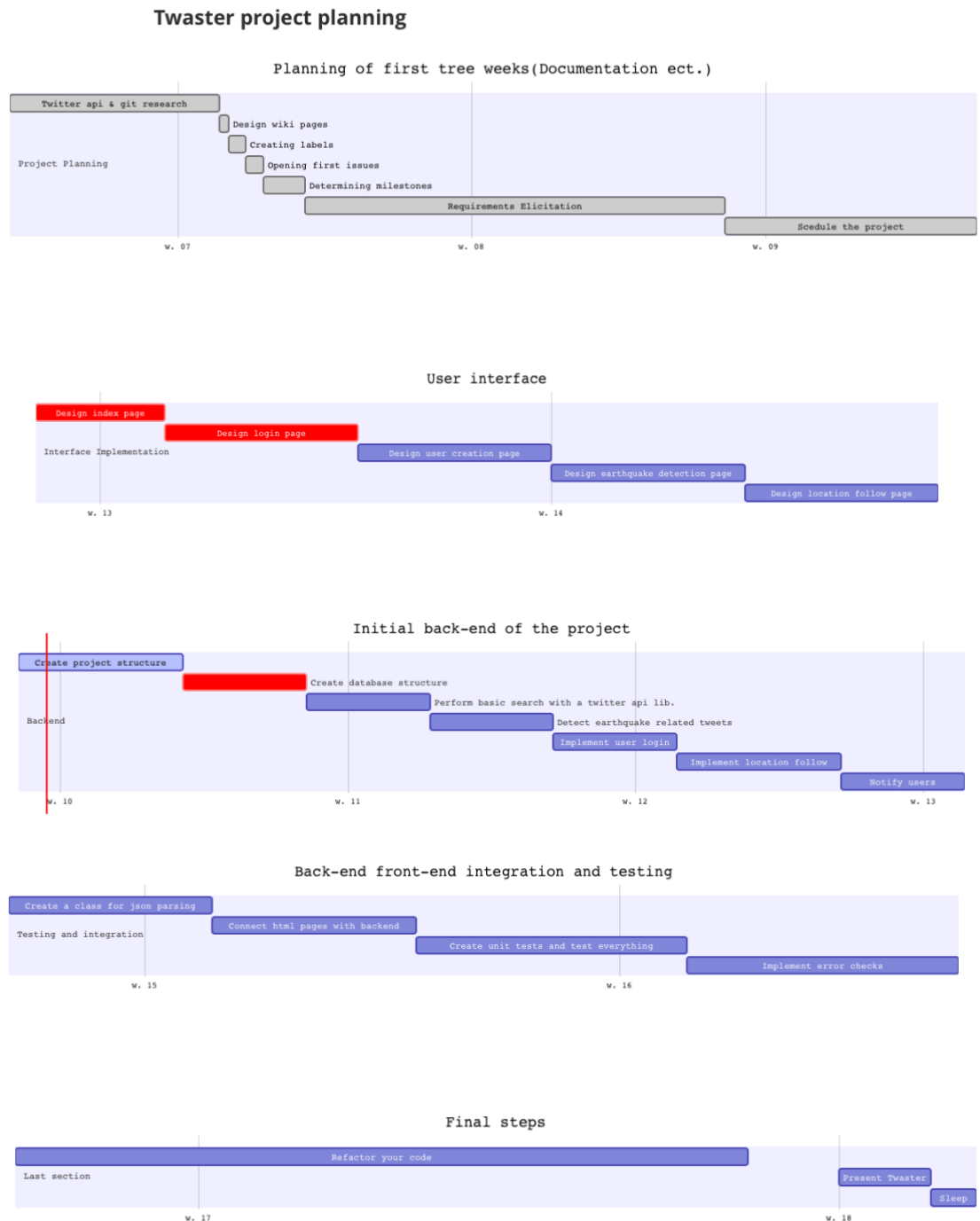
- A user should be able to create an account at twasterapp.com.
- A user should be able to login with his credentials.
- A user should be able to select cities to listen.
- A user must be informed when an earthquake is detected in listed locations of himself. (Email notification for now)
- A user should be able reset his password.
- A user should be able to see locations visually by a map.
- A user should be able to logout from system.

System requirements:

- A database system must record many information about detected tweets, event related tweets.
- Same system must record all information about user credentials.
- A highly supported and available open source framework must be used. (Django in this case)
- A reasonable user interface must be created to interact with backend.
- Background scripts must be created for earthquake detection.
- System should be deployed.

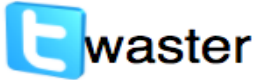
5.3. PLANNING PHASE

In planning, I have used an JS library and created a plan using markdown language. Name of the library is mermaid.js Here are the details.



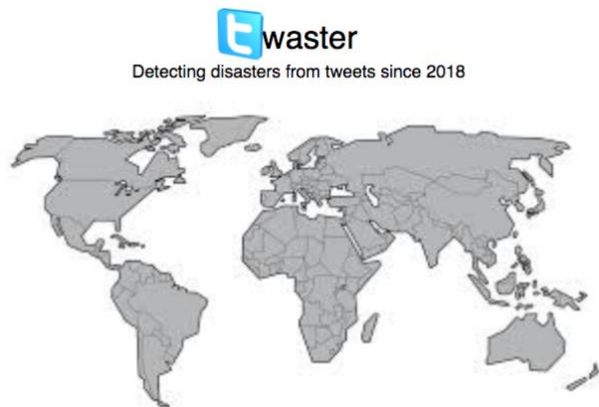
5.3. Mock-ups

Mock-ups created by using <https://mockups.com>.



Detecting disasters from tweets since 2018

▼ Location	▼ Detection Time	▼ Actual Time
Nagasaki	11.14.18 03:42	11.14.18 03:42
St. Petersburg	11.14.18 05:59	11.14.18 04:00
Manisa	11.14.18 07:59	11.14.18 08.01
Nagasaki	10.14.18 03:42	11.14.18 03:42
St. Petersburg	10.14.18 05:59	11.14.18 04:00
Manisa	10.14.18 07:59	11.14.18 08.01
Nagasaki	09.14.18 03:42	11.14.18 03:42
St. Petersburg	09.14.18 05:59	11.14.18 04:00
Manisa	09.14.18 07:59	11.14.18 08.01
Nagasaki	08.14.18 03:42	11.14.18 03:42
St. Petersburg	08.14.18 05:59	11.14.18 04:00
Manisa	08.14.18 07:59	11.14.18 08.01
Nagasaki	07.14.18 03:42	11.14.18 03:42
St. Petersburg	07.14.18 05:59	11.14.18 04:00
Manisa	07.14.18 07:59	11.14.18 08.01
Nagasaki	06.14.18 03:42	11.14.18 03:42
St. Petersburg	06.14.18 05:59	11.14.18 04:00



SS 3 login page

Last 3 earthquake		
▼ Location	▼ Detection Time	▼ Actual Time
Nagasaki	11.14.18 03:42	11.14.18 03:42
St. Petersburg	11.14.18 05:59	11.14.18 04:00
Manisa	11.14.18 07:59	11.14.18 08.01



SS 4 location follow page

5.4. Use Case Scenarios

A daughter far away

Sinem Truncgil is a housewife. She has a daughter which is currently a University student. Her name is Selin and she has got a scholarship from Tokyo University. Sinem heard that Japan is in destinations of tons of earthquakes, typhoons and natural disasters occasionally. She wants to be sure that her daughter is safe.

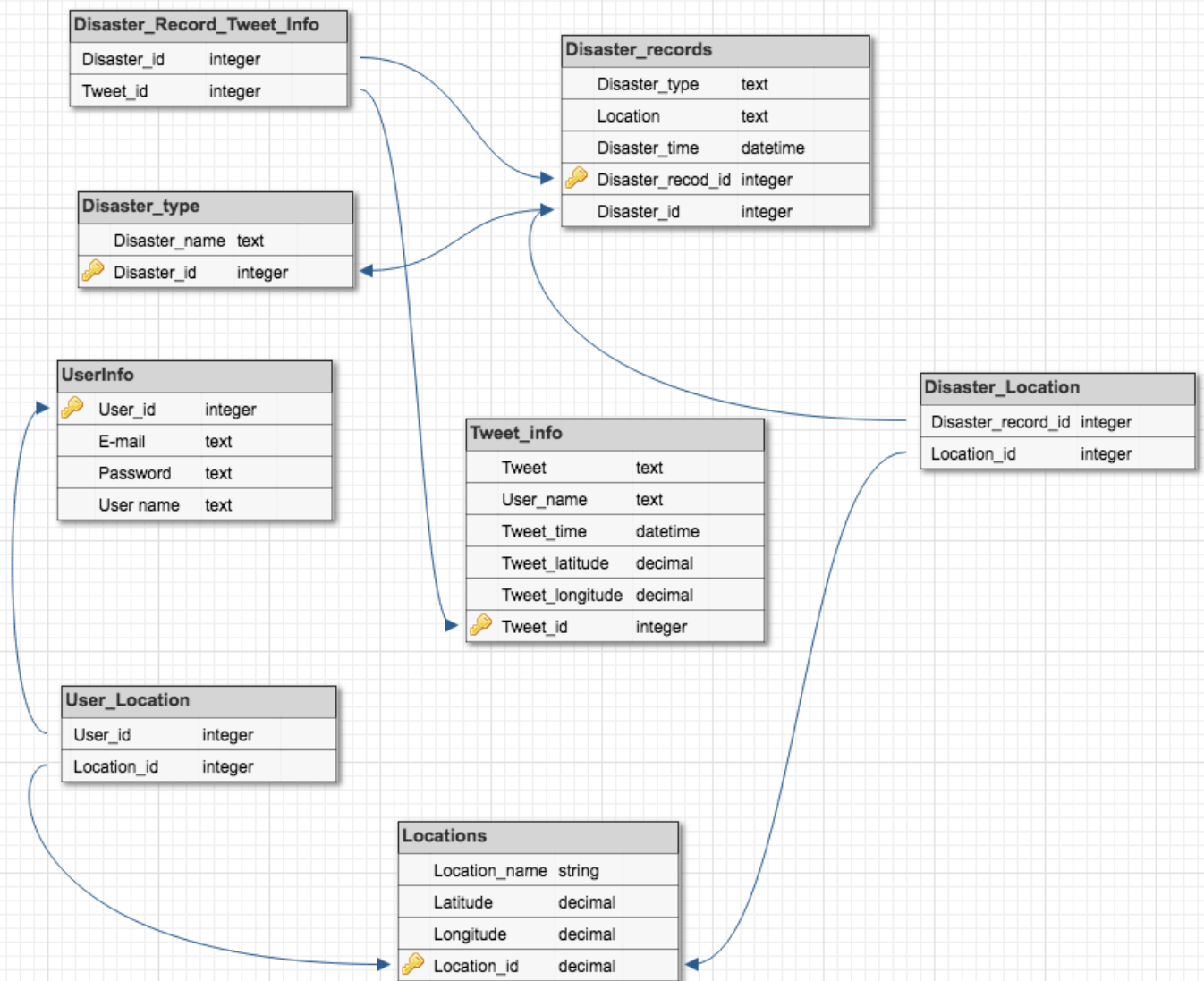
1. Sinem login with her e-mail and pass to the site.
2. She picks multiple locations to follow from world map or a list of cities in the world. In her case she chooses Tokyo, Nagoya and Iwaki. (The possible places where her girl can travel within her budget)
3. Application start to listen that locations and informs Sinem when a disaster occurs in given Cities.
4. She continues her daily routine.

- ## 5.5. Design Phase

```
classDiagram
    class DisasterRecord {
        id AutoField
        disaster_type ForeignKey (id)
        location ForeignKey (id)
        record_time TextField
        record_date DateField
    }
    class Tweet {
        id AutoField
        location TextField
        tweet TextField
        tweet_latitude DecimalField
        tweet_longitude DecimalField
        tweet_time DateTimeField
        user_name TextField
    }
    class User {
        id AutoField
        location ForeignKey (id)
        user ForeignKey (id)
    }
    class Location {
        id AutoField
        altitude DecimalField
        city CharField
        country CharField
        latitude DecimalField
        longitude DecimalField
    }
    class Group {
        id AutoField
        name CharField
    }
    class Permission {
        id AutoField
        content_type ForeignKey (id)
        codename CharField
        name CharField
    }
    class AbstractUser {
        <<AbstractUser>>
        id AutoField
        date_joined DateTimeField
        email EmailField
        first_name CharField
        is_active BooleanField
        is_staff BooleanField
        is_superuser BooleanField
        last_login DateTimeField
        last_name CharField
        password CharField
        username CharField
    }
    class AbstractBaseSession {
        <<AbstractBaseSession>>
        session_key CharField
        expire_date DateTimeField
        session_data TextField
    }
    class ContentType {
        id AutoField
        app_label CharField
        model CharField
    }
    class DisasterRecord_Tweet_Info {
        id AutoField
        disaster_record ForeignKey (id)
        tweet ForeignKey (id)
    }
    class DisasterRecord_Location {
        id AutoField
        disaster_record ForeignKey (id)
        location ForeignKey (id)
    }
    class User_Location {
        id AutoField
        user ForeignKey (id)
        location ForeignKey (id)
    }
    class LogEntry {
        id AutoField
        content_type ForeignKey (id)
        user ForeignKey (id)
        action_flag CharField
        action_time DateTimeField
        change_message TextField
        object_id TextField
        object_repr CharField
    }
    class DisasterType {
        id AutoField
        disaster_name TextField
    }
    class AbstractBaseUser {
        <<AbstractBaseUser>>
        id AutoField
        date_joined DateTimeField
        email EmailField
        first_name CharField
        is_active BooleanField
        is_staff BooleanField
        is_superuser BooleanField
        last_login DateTimeField
        last_name CharField
        password CharField
        username CharField
    }
    class PermissionsMixin {
    }
    class AbstractBaseUser {
    }
    class ContentType {
    }

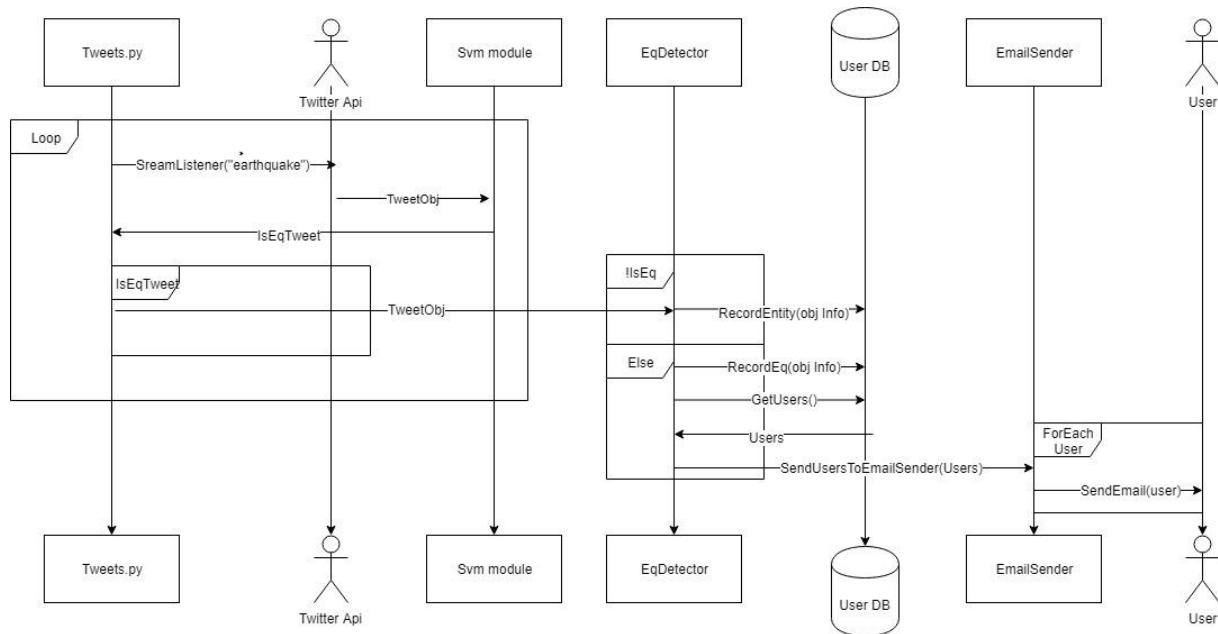
    DisasterRecord --> Tweet : Disaster_record_tweet_info
    DisasterRecord --> Location : Disaster_record_location
    User --> Location : User_location
    User --> LogEntry : User_logentry
    Group --> Permission : Permissions_group
    AbstractUser --> AbstractBaseUser : abstract inheritance
    AbstractBaseSession --> AbstractBaseUser : abstract inheritance
    ContentType --> AbstractBaseUser : abstract inheritance
    ContentType --> PermissionsMixin : abstract inheritance
    ContentType --> AbstractBaseUser : abstract inheritance
```

Database Design

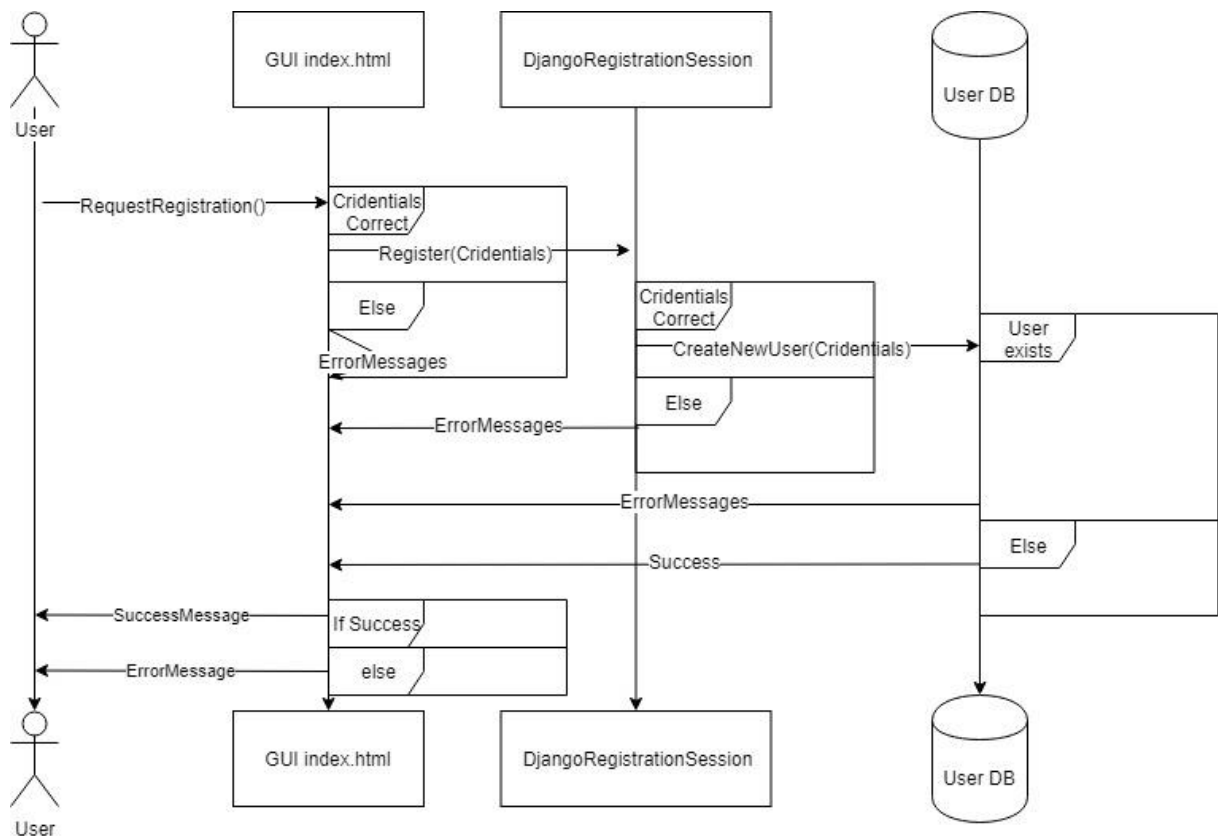


5.5.1 Sequence Diagrams

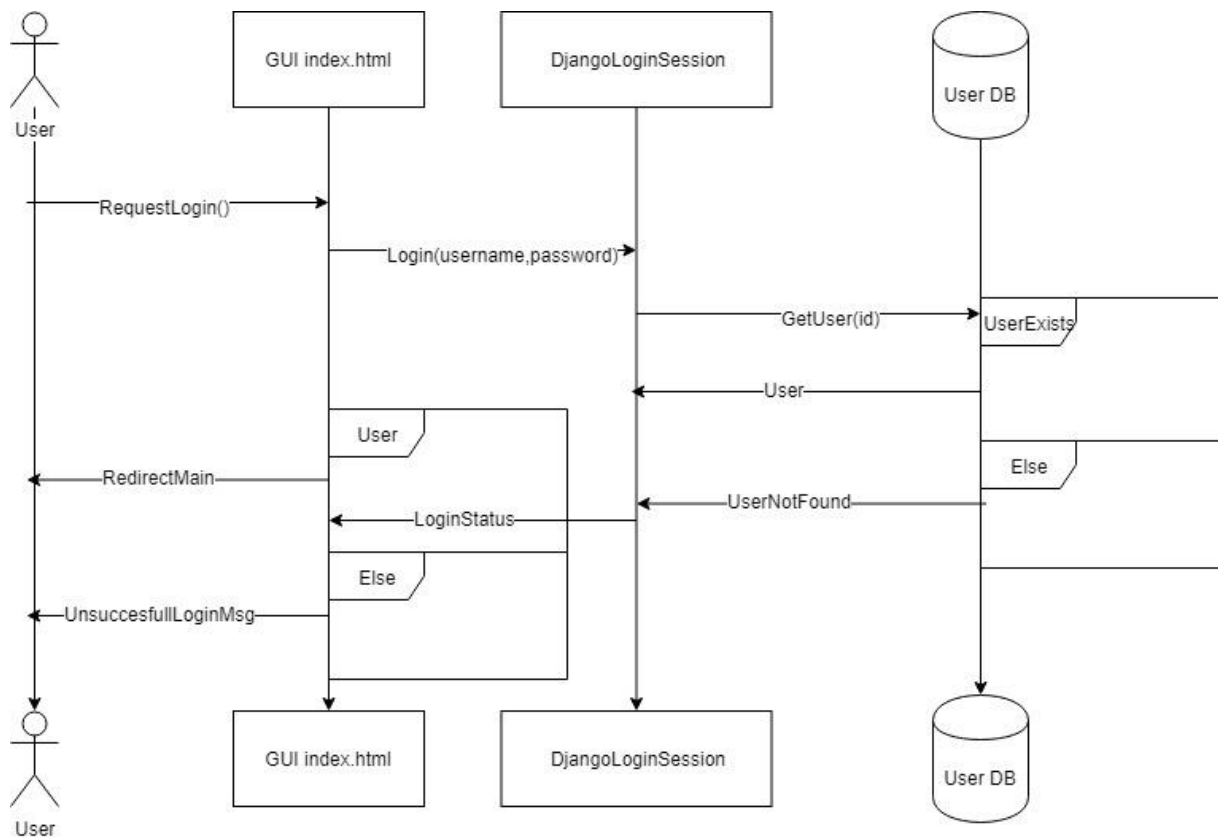
Earthquake Detection



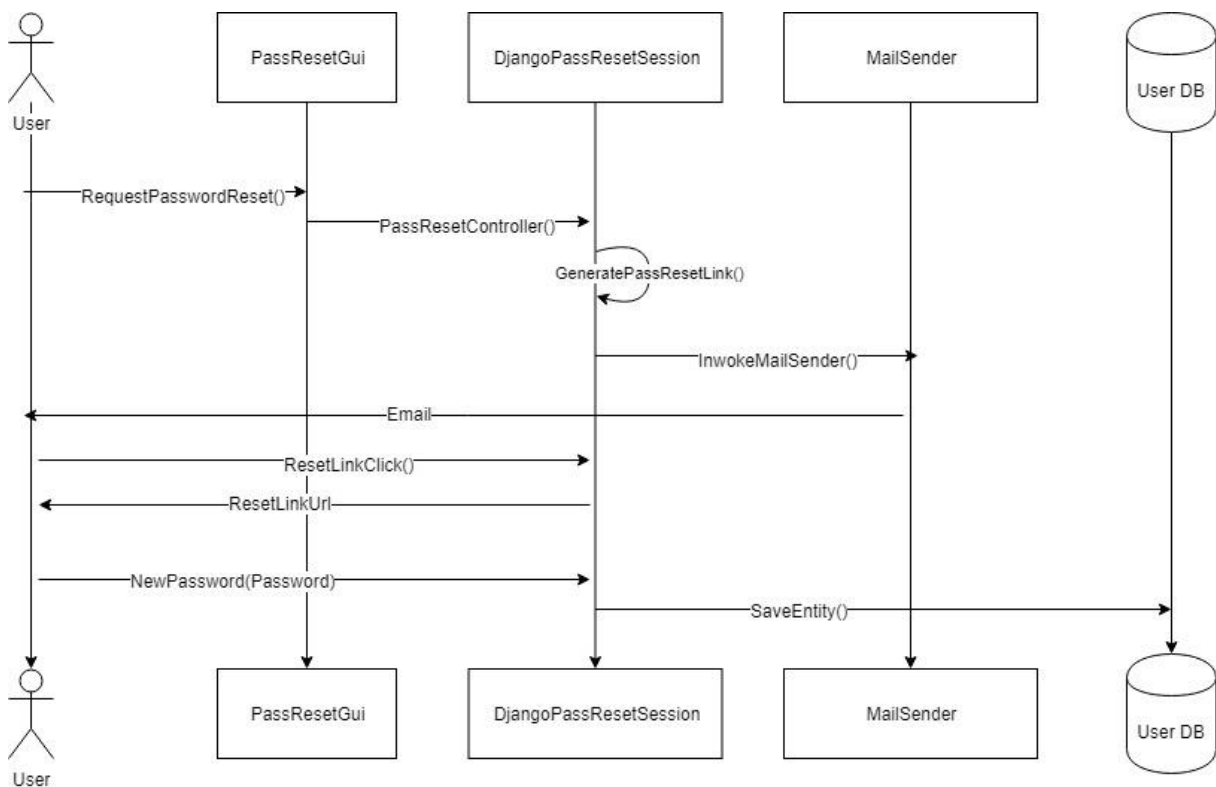
User Registration



User Login



Password Reset




6. DEPLOYMENT

As a hosting platform, I choose pythonanywhere.com because ease of deployment, free SSL certificate support, it's specified for python usage and it is relatively cheap according to other competitors.

Here is the address which you can visit the deployed site and create an account:
<https://www.twasterapp.com/TweetUtils/>

TWASTER

Login

waster


Detecting earthquakes from twitter since 2018

Show entries

Search:

Record No	City	Country	Detection Time
1	Riverside	United States	7:14 a.m. May 12, 2018
2	Aberdeen	United Kingdom	3:42 a.m. May 12, 2018
3	Matsumoto	Japan	1:36 a.m. May 12, 2018
4	Ormoc	Philippines	11:58 a.m. May 10, 2018
5	Charlotte	United States	9:03 p.m. May 9, 2018
6	Tulsa	United States	2:13 p.m. May 9, 2018
7	Lahore	Pakistan	12:09 p.m. May 9, 2018
8	Delhi	India	11:29 a.m. May 9, 2018
9	Lahore	Pakistan	10:55 a.m. May 9, 2018
10	Islamabad	Pakistan	10:44 a.m. May 9, 2018
11	Santa Ana	United States	11:55 a.m. May 8, 2018
12	San Diego	United States	11:52 a.m. May 8, 2018
13	Riverside	United States	11:50 a.m. May 8, 2018

SS 5 index page

tweetlistener

Share with others

```
google maps = long : -91.530168 lat : 41.661128
318 : When the earth is shaken,
With it's (final) earthquake.
Al_Quran; 99:1
google maps = long : 69.345116 lat : 30.375321
321 : hit wonder full earthquake
google maps = long : -81.250195 lat : 42.980353
324 : Like an earthquake
https://t.co/5JPSaTD82W
google maps = long : -3.70379 lat : 40.416775
325 : Not an earthquake request.
with place = long : -119.172179 lat : 35.346902
328 : Was that a earthquake or am I trippen
google maps = long : -117.844296 lat : 33.640495
330 : it's time for-foggy-earthquake
google maps = long : -75.197534 lat : 40.986937
332 : We just felt an earthquake in HPP 1154pm
google maps = long : -111.694648 lat : 40.296898
334 : yo, did anyone else feel that earthquake?
google maps = long : -155.665857 lat : 19.542915
336 : #earthquake
3.5-3.9+
Two
Outside India quake
google maps = long : 81.871606 lat : 25.537687
337 : Did you feel the earthquake.
google maps = long : -117.435048 lat : 34.092233
340 : Causing an earthquake, breaking the ground !!
google maps = long : -95.418048 lat : 29.732552
342 : Shake it like an earthquake. #ruebot
google maps = long : 139.703549 lat : 35.69384
344 : lets play: earthquake or anxiety shivers JAPAN VERSION
google maps = long : 140.446794 lat : 36.341811
347 : Dirty Italian should've died in that earthquake
google maps = long : 38.998105 lat : 35.959411
348 : Tfw you wake up to thunder and think it's an earthquake #CainPA
google maps = long : -122.241636 lat : 37.765206
```

SS 6 Long running script for earthquake detection

7. CONCLUSION & FUTURE WORK

As a part of Software Development Practice lesson, I have developed a web page which listens all tweets around the globe and detects earthquakes based on that information. This is a very interesting example of power of social media. So far my detection success rate is above %90 with little improvements I believe the success rate easily could increase above to %98.

Also about the application it seems useful but I would not recommend using it as a main source of earthquake detection information yet because tweet users are not sitting around every corner of the world. But as a side kick it is great to use it. Especially if you are living in a place where twitter usage is high and in a country where English is used as a mother tongue.

As stated above one of the major issue is to add more languages for earthquake detection. We can detect more earthquakes and more precisely. Also, it would be great to fetch more data to increase classification in SVM algorithm. It is a painful process because Twitter API won't allow you to fetch data more than 1 month old but there is work arounds.

As I last word, I strongly believe human sensors as twitter users will be more important in near future for event detection. This project is a beginning for harnessing of power in social media in a good manner and has a lot of potential to improve.

8. REFERENCES

- [1] <http://www.bgs.ac.uk/discoveringGeology/hazards/earthquakes>
- [2] Twitter earthquake detection: earthquake monitoring in a social world Paul S. Earle, Daniel C. Bowden, Michelle Guy
- [3] Tweet Analysis for Real-Time Event Detection and Earthquake Reporting System Development Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo