



**Question/Answering System for Diseases**

## **Team 12**

- ❖ Gupta, Arunit – 11
- ❖ Patel, Marmikkumar Navinchandra – 31
- ❖ McDuff, Luke Joseph – 21
- ❖ Ejjirothu, Manoj Prabhakar – 7

## Table of Contents

1. Introduction .....	4
2. Project Goals and Objectives .....	5
2.1 Motivation.....	5
2.2 Objective .....	5
2.3 Expected Outcomes .....	5
3. Project Domain & Datasets .....	6
3.1 Project Domain .....	6
3.2 Dataset Links .....	6
4. Features Implementation .....	7
4.1 Word Count.....	7
4.2 NLP Processing .....	8
4.3 Information Retrieval/Extraction Technologies.....	10
4.4 Word2Vec .....	11
4.5 Word Net.....	13
4.6 Topic Discovery Using LDA .....	15
4.7 Feature Vector Generation .....	18
4.8 K- Means .....	19
5. Implementation Specification.....	21
5.1 Software Architecture Diagram .....	21
5.2 Class Diagram .....	22
5.3 Sequence Diagram .....	23
5.4 Workflow.....	24
5.5 User Interface Wire frame .....	25
5.6 Expected Output .....	26
5.7 Existing services/features used.....	26
6. Project Management .....	27
6.1 Contribution of each member .....	27

6.2 Zen hub and GitHub URL/Statistics .....	27
6.3 Concerns/Issues .....	29
6.4 Future Work .....	29
7. References .....	30

# 1. Introduction

## Geographical Question and Answering system for Diseases

Twitter is a vast source of unstructured data that when correctly extracted on a large scale can reveal valuable information. One of the more interesting aspects of the data contained in one Twitter tweet is its meta-data. The meta-data for a tweet contains various information, but what is of interest for this project is the geolocation meta-data specifying the longitude and latitude for a particular tweet. Knowing the geolocation of a tweet will help us to understand where a tweet is coming from and make decisions based on geographical densities of similar tweets.

Imagine an outbreak in the United States of a particular strain of flu occurs, for example we will use H1N1. As researchers, we want to be able to determine where in the United States the H1N1 outbreak is occurring and what areas are likely to be the most affected. By observing the trends on twitter related to H1N1 and being able to accurately conclude the H1N1 disease content of the tweet, we can determine the geographical relationship of the disease, showing clusters of activity. This information can help researchers to better understand the spread of particular diseases in certain regions as well as provide health officials real-time information regarding the spread of the disease.

This information can be displayed effectively in a heat map presented to the user for an answer. The user will be able to interact and query for geographical disease answers via a website. They will be able to enter a query regarding their geographic question and the solution will be a heat map. The scope of this project will be narrow, but will provide new information and answers that cannot be found in a dictionary or database.

## 2. Project Goals and Objectives

### 2.1 Motivation

In our world today, information moves at a very high rate of speed. Within a few days information may be outdated or irrelevant. With information sources as Twitter we are able to get info immediately. Even though information is immediate, it is difficult to filter and provide meaningful representations of the data. Our motivation is to provide the user the ability to retrieve geographical information related to diseases the moment an event occurs and generate a representation of that information that easy to understand and process.

### 2.2 Objective

To design a system which can take a question from a user and give an answer as output by applying natural language and machine learning principles and display the information in a structured format.

### 2.3 Expected Outcomes

A webpage accessible by users across the world having the ability to ask a question related to geography of diseases, and get a response on the webpage in the form of a heat map.

## 3. Project Domain & Datasets

### 3.1 Project Domain

Healthcare - Consisting of Diseases, geography of diseases, health issues, drug information, trending viruses, allergies etc.

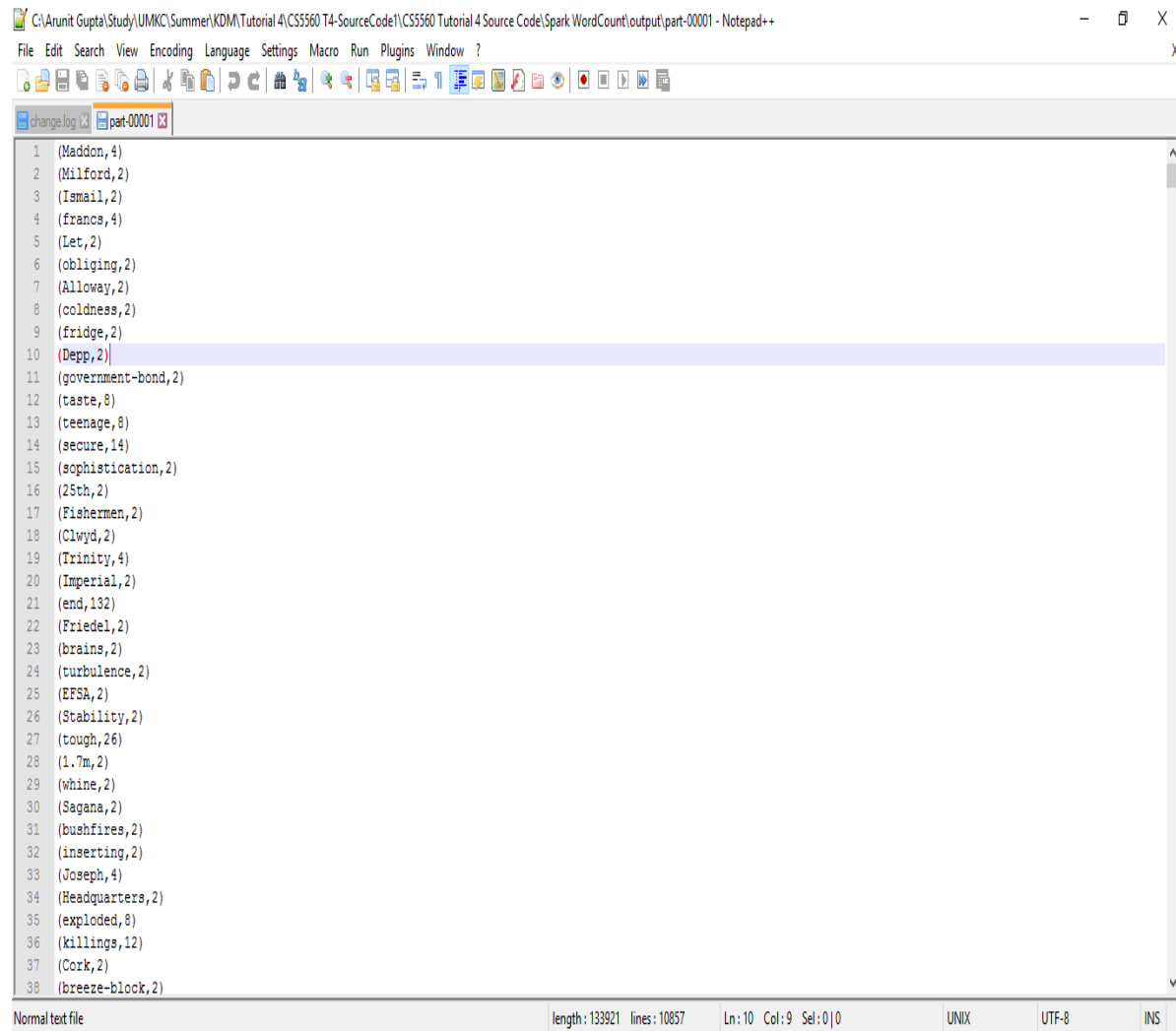
### 3.2 Dataset Links

- ❖ <https://dev.twitter.com/overview/api> - Collected 200K tweets on health and Diseases.
- ❖ <http://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2>
- ❖ <http://www.statmt.org/lm-benchmark/1-billion-word-language-modeling-benchmark-r13output.tar.gz>
- ❖ <http://ebiquity.umbc.edu/redirect/to/resource/id/351/UMBC-webbase-corpus>
- ❖ <http://blog.wolframalpha.com/category/health-med/>
- ❖ <http://archive.ics.uci.edu/ml/datasets/Hepatitis>
- ❖ <http://www.cdc.gov/>
- ❖ <http://www.cdc.gov/DataStatistics/>
- ❖ <http://www.cdc.gov/hepatitis/Statistics/index.htm>

## 4. Features Implementation

### 4.1 Word Count

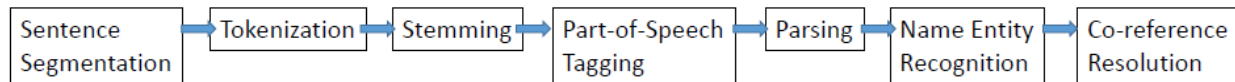
Word count program searches for the input file and reads the file sentence by sentence, it produces an output file with words and count of those words in the corpus of data



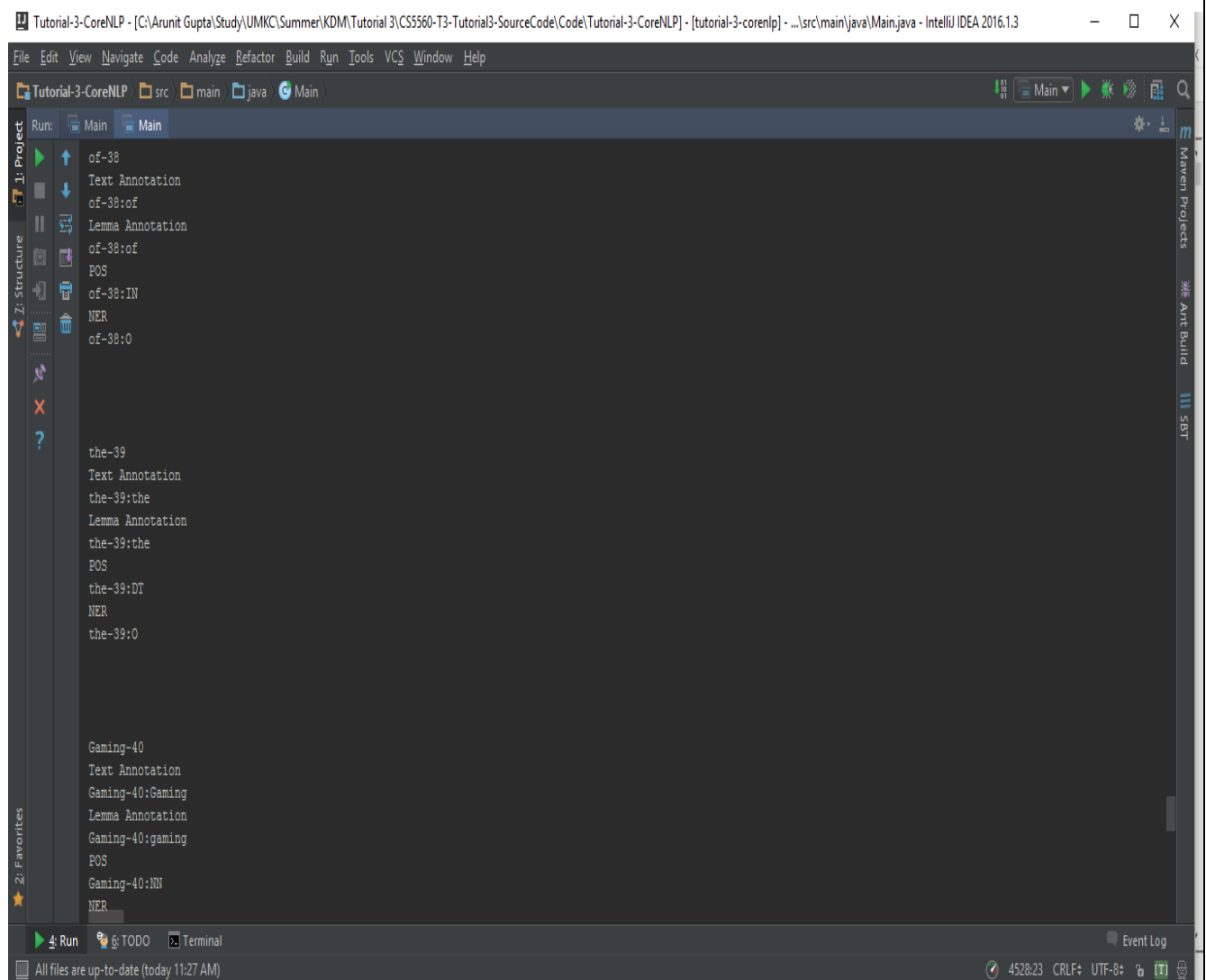
```
1 (Maddon,4)
2 (Milford,2)
3 (Ismail,2)
4 (frances,4)
5 (Let,2)
6 (obliging,2)
7 (Alloway,2)
8 (coldness,2)
9 (fridge,2)
10 (Depp,2)
11 (government-bond,2)
12 (taste,8)
13 (teenage,8)
14 (secure,14)
15 (sophistication,2)
16 (25th,2)
17 (Fishermen,2)
18 (Clwyd,2)
19 (Trinity,4)
20 (Imperial,2)
21 (end,132)
22 (Friedel,2)
23 (brains,2)
24 (turbulence,2)
25 (EFSA,2)
26 (Stability,2)
27 (tough,26)
28 (1.7m,2)
29 (whine,2)
30 (Sagana,2)
31 (bushfires,2)
32 (inserting,2)
33 (Joseph,4)
34 (Headquarters,2)
35 (exploded,8)
36 (killings,12)
37 (Cork,2)
38 (breeze-block,2)
```

## 4.2 NLP Processing

NLP is the branch of computer science focused on developing systems that allow computers to communicate with people using everyday language.



### Lemmatization





## POS

Tutorial-3-CoreNLP - [C:\Arunit Gupta\Study\UMKC\Summer\KDM\Tutorial 3\CS5560-T3-Tutorial3-SourceCode\Code\Tutorial-3-CoreNLP] - [tutorial-3-corenlp] - ...src\main\java\Main.java - IntelliJ IDEA 2016.1.3

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

Tutorial-3-CoreNLP src main java Main

Run: Main Main

1: Project

2: Structure

2: Favorites

```

(ROOT (S (ADVP (RB Now)) (NP (DT the) (NN gambler)) (VP (MD will) (VP (VB have) (S (VP (TO to) (VP (VB convince) (NP (DT a) (NN judge)) (SBAR (IN that) (S (NP (NP (DT the) (NN
-> have/VB (root)
-> Now/RB (advmod)
-> gambler/NN (nsubj)
-> the/DT (det)
-> will/MD (aux)
-> convince/VB (xcomp)
-> to/TO (mark)
-> judge/NN (dobj)
-> a/DT (det)
-> provision/NN (ocomp)
-> that/IN (mark)
-> cheque/NN (nsubj)
-> the/DT (det)
-> gave/VBD (acl:relcl)
-> he/PRP (nsubj)
-> Club/NNP (dobj)
-> Aspinall/NNP (nmod:poss)
-> 's/POS (case)
-> London/NNP (nmod:in)
-> in/IN (case)
-> ,/, (punct)
-> claimed/VBD (acl:relcl)
-> which/WDI (dobj)
-> he/PRP (nsubj)
-> dated/VBN (ocomp)
-> had/VBD (aux)
-> not/RB (neg)
-> been/VBN (auxpass)
-> ,/, (punct)
-> was/VBD (cop)
-> an/DT (det)
-> unlawful/JJ (amod)

```

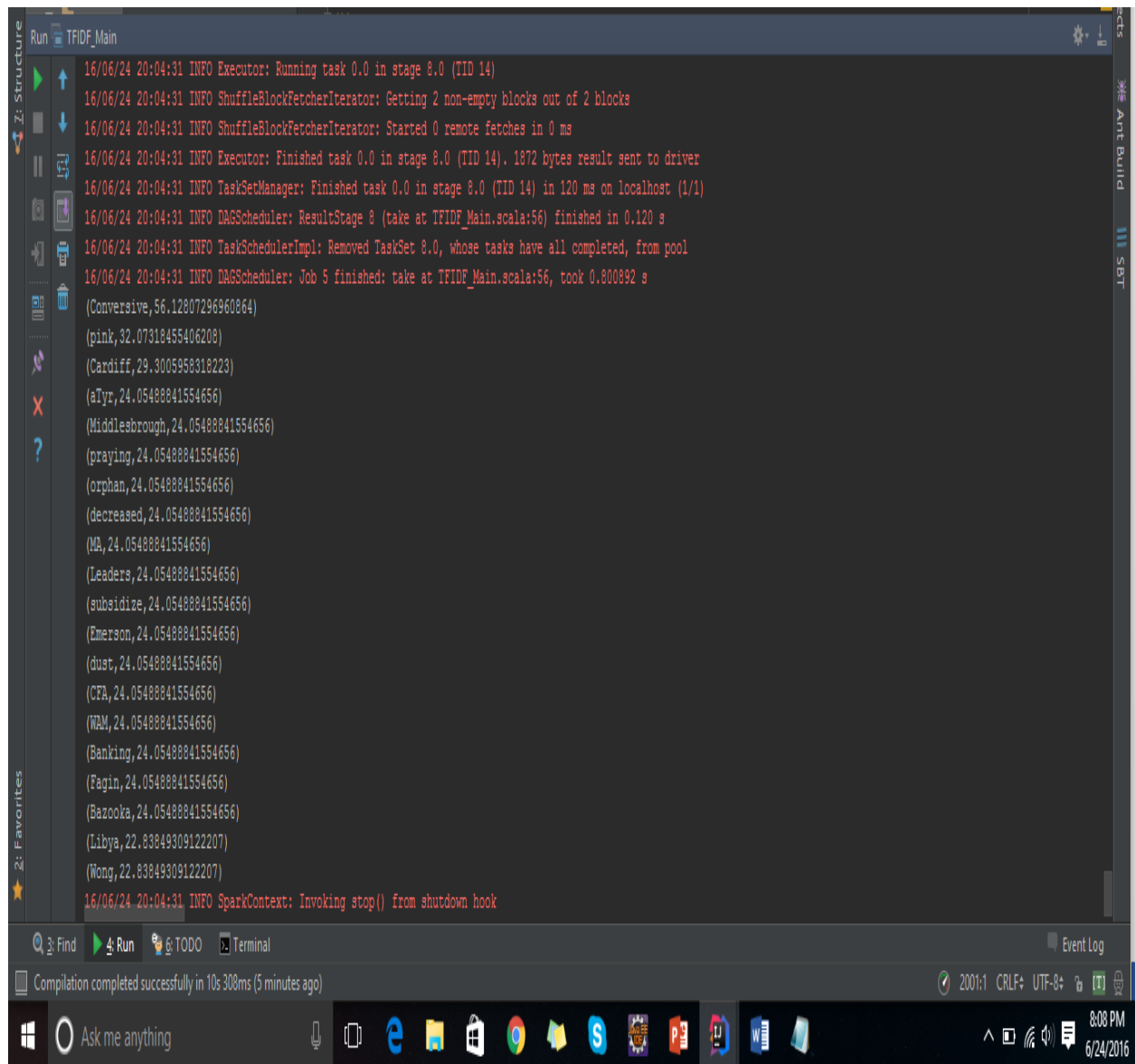
Run TODO Terminal Event Log

All files are up-to-date (today 11:27 AM) 4038:17 CRLF+ UTF-8+

## 4.3 Information Retrieval/Extraction Technologies

TFIDF, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.

It displays the word with the frequency of it in the corpus.



```

Run TFIDF_Main
16/06/24 20:04:31 INFO Executor: Running task 0.0 in stage 8.0 (TID 14)
16/06/24 20:04:31 INFO ShuffleBlockFetcherIterator: Getting 2 non-empty blocks out of 2 blocks
16/06/24 20:04:31 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
16/06/24 20:04:31 INFO Executor: Finished task 0.0 in stage 8.0 (TID 14). 1872 bytes result sent to driver
16/06/24 20:04:31 INFO TaskSetManager: Finished task 0.0 in stage 8.0 (TID 14) in 120 ms on localhost (1/1)
16/06/24 20:04:31 INFO DAGScheduler: ResultStage 8 (take at TFIDF_Main.scala:56) finished in 0.120 s
16/06/24 20:04:31 INFO TaskSchedulerImpl: Removed TaskSet 8.0, whose tasks have all completed, from pool
16/06/24 20:04:31 INFO DAGScheduler: Job 5 finished: take at TFIDF_Main.scala:56, took 0.800892 s
(Conversive,56.12807296960864)
(pink,32.07318455406208)
(Cardiff,29.3005958318223)
(aTyr,24.05488841554656)
(Middlesbrough,24.05488841554656)
(praying,24.05488841554656)
(orphan,24.05488841554656)
(decreased,24.05488841554656)
(MA,24.05488841554656)
(Leaders,24.05488841554656)
(subsidize,24.05488841554656)
(Emerson,24.05488841554656)
(dust,24.05488841554656)
(CFA,24.05488841554656)
(WAM,24.05488841554656)
(Banking,24.05488841554656)
(Pagin,24.05488841554656)
(Bazooka,24.05488841554656)
(Libya,22.83849309122207)
(Wong,22.83849309122207)
16/06/24 20:04:31 INFO SparkContext: Invoking stop() from shutdown hook

Compilation completed successfully in 10s 308ms (5 minutes ago)
2001:1 CRLF+ UTF-8+
8:08 PM
6/24/2016

```

## 4.4 Word2Vec

Word2Vec computes distributed vector representation of words. The main advantage of the distributed representations is that similar words are close in the vector space, which makes generalization to novel patterns easier and model estimation more robust.

First NLP processing is done on input which includes processes like tokenization, lemmatization, POS tagging. After that Stop Words are removed from dataset and then ngram processing is applied on input. After ngram processing top 20 TFIDF words are selected. In the last step, 3 synonyms for selected top 20 TFIDF words are selected. In the output given above, first is schema of ngram processing. After that top 6 TFIDF words are printed with their TFIDF. In the last, 3 synonyms for each of these words are printed with cosine Similarity Score.

### Output:

```
root
|-- labels: integer (nullable = false)
|-- sentence: string (nullable = true)
|-- words: array (nullable = true)
|   |-- element: string (containsNull = true)
|-- filteredWords: array (nullable = true)
|   |-- element: string (containsNull = true)
|-- ngrams: array (nullable = true)
|   |-- element: string (containsNull = false)
```

```
()
```

```
(pd,72.20820018539699)
```

```
(dxs,54.75862726423568)
```

```
(dmd,33.94648454931014)
```

```
(theta,27.032740041837865)
```

```
(yac,23.56700413903814)
```

```
(gd,23.56700413903814)
```

pd :

[character,0.9989249542273985]

[deposition,0.9985913150891267]

[ema,0.9985725473521622]

dxs :

[pdf,0.9994259508897761]

[spread,0.9990591262599753]

[medium,0.9987120247218979]

dmd :

[cardiovascular,0.9995456261106753]

[pws,0.9994482423773895]

[shah,0.9993167751788441]

theta :

[epistasis,0.9998306768961894]

[normally,0.9998038685282771]

[weird,0.9997854770386241]

yac :

[endocytic,0.9995076438544604]

[ester,0.9992897088344618]

[deltg,0.9992195317514144]

gd :

[neurodevelopmental,0.9997400737808784]

[editing,0.9996893399532325]

[satisfaction,0.9996790665181463]

## 4.5 Word Net

WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms, each expressing a distinct concept

WordNet displays the definition of the particular word, synonyms of each word. Hyponyms and Hypernyms for the words.

### **Output:**

Definitions for health:

a healthy state of wellbeing free from disease the general condition of body and mind

### **Synonyms for tired (pos: a)**

all in

a weary

banal

beat

blear

blear-eyed

bleary

bleary-eyed

bored

burned-out

### **Hyponyms for virus:**

arbovirus

arbovirus

bacteriophage

phage

plant virus

animal virus

slow virus

tumor virus

vector

**Hypernyms for virus:**

microorganism

micro-organism

infectious agent

infective agent

representation

delegacy

agency

malevolent program

## 4.6 Topic Discovery Using LDA

LDA is a Statistical model which grasps the similar words in documents and clubs them into a topic. The topics produced by the LDA is not strongly defined by the words in it, all the words in a given topic are tagged with the probability scores which allows us to know the extent to which the particular word belongs to the Topic.

The Knowledge base for this project belongs to the Health Care Domain - which consists of the information regarding the diseases virus, drugs etc. The results of the LDA code for the above mentioned data sets is below.

Processing Time

Corpus summary:

```
Training set size: 1838282 documents
Vocabulary size: 262144 terms
Training set size: 25425333 tokens
Preprocessing time: 7970.322794073 sec
```

The above data represents the total summary of the data which was given as input the LDA code. It calculates the documents in each file and calculates the number of tokens which are to be categorized into groups

Finished training LDA model. Summary:

```
Training time: 758.407606948 sec
Training data average log likelihood: -119.12721279634076
```

The above data represents the time taken by the LDA model to categorize the given input data into the specified number of topics

### Topic Output

The code was run in order to group the tokens into 3 different topics ( $k=3$ ). Below is the sample output of all the three topics. The output shows the top weighted words which belong to the particular topic. The words may be repeated into multiple topic but they are categorized based on the weight at which they

belong to a particular topic. The multinomial distribution parameter (weighted score) is allocated to each word in the document from the Dirichlet distribution.

### **Topic 0**

TOPIC\_0;pneumonia;8.546742342173153E-6  
TOPIC\_0;zika;8.546635162343083E-6  
TOPIC\_0;quo;8.544962433280642E-6  
TOPIC\_0;utilization;8.54388798531218E-6  
TOPIC\_0;lohan;8.542477615504052E-6  
TOPIC\_0;sprawling;8.54189567221355E-6  
TOPIC\_0;embattled;8.541159906496557E-6  
TOPIC\_0;seminar;8.539338027311975E-6  
TOPIC\_0;divisive;8.537907695228484E-6  
TOPIC\_0;apparel;8.5374808905532E-6  
TOPIC\_0;indirect;8.537070817115574E-6  
TOPIC\_0;real-life;8.534108828951073E-6  
TOPIC\_0;validate;8.530082949570476E-6  
TOPIC\_0;ego;8.528871874541697E-6  
TOPIC\_0;napolitano;8.528844718675242E-6

### **Topic 1**

TOPIC\_1;news;7.389598406650746E-4  
TOPIC\_1;share;7.388731701000441E-4  
TOPIC\_1;open;7.387333347891583E-4  
TOPIC\_1;attack;7.38215407010838E-4  
TOPIC\_1;rise;7.357389221444835E-4  
TOPIC\_1;offer;7.340158605047407E-4  
TOPIC\_1;family;7.315727982900956E-4  
TOPIC\_1;security;7.267220323694219E-4  
TOPIC\_1;monday;7.256736653433311E-4  
TOPIC\_1;financial;7.248604331935932E-4  
TOPIC\_1;wednesday;7.230609958056925E-4  
TOPIC\_1;increase;7.215121360961194E-4  
TOPIC\_1;area;7.214202500773856E-4  
TOPIC\_1;begin;7.163421379115239E-4  
TOPIC\_1;follow;7.156402898848389E-4  
TOPIC\_1;deal;7.143427641696115E-4  
TOPIC\_1;friday;7.134515110513112E-4  
TOPIC\_1;don;7.129673315607219E-4  
TOPIC\_1;kill;7.12581046207116E-4  
TOPIC\_1;money;7.103314559704194E-4  
TOPIC\_1;thursday;7.09468637873985E-4



**Topic 2**

TOPIC\_2;growth;3.126299027308623E-4  
TOPIC\_2;central;3.125294472762388E-4  
TOPIC\_2;development;3.123319544389857E-4  
TOPIC\_2;reduce;3.1203330357669533E-4  
TOPIC\_2;hard;3.116780319585157E-4  
TOPIC\_2;possible;3.1095236346100915E-4  
TOPIC\_2;role;3.109326064246385E-4  
TOPIC\_2;loan;3.103130226855498E-4  
TOPIC\_2;form;3.088434315349157E-4  
TOPIC\_2;network;3.088145815280927E-4  
TOPIC\_2;key;3.088017551801666E-4  
TOPIC\_2;accuse;3.0840250250604957E-4  
TOPIC\_2;medical;3.0783239781282653E-4  
TOPIC\_2;position;3.07823469479574E-4

## 4.7 Feature Vector Generation

Feature vector is a kind of an object which represents a set of documents. The Feature Vector generation depends on the term frequencies. The information retrieval from the Documents provided through the TF-IDF method is done in order to know the importance of a word. The words are allocated with the probability scores and represented in the Vector space with a matrix. The Vector space which has all the feature vectors represents the entire clan of the documents. Feature Vector Generation for our Input Set of Documents is as below.

Output represents the Feature vectors generated from the pipeline. Taking the Term Frequency into account, get the top TF words and through the Word2VEC the above Feature Vectors are generated long with their Vector Space Scores.

### Output:

```
(category,2297.932291433732)
(specificdisease,660.2817683387467)
(mutation,345.2030738830825)
(modifier,288.52461184927324)
(gene,284.28488437430326)
(diseaseclass,174.2751135763978)
(zika,170.43007612859284)
(pd,128.21723156526366)
(family,126.52239359515694)
(analysis,85.01769305071392)
(deficiency,81.67053978100077)
(chromosome,79.8853913704871)
(deletion,76.98452520340237)
(cancer,74.30680258763185)
(normal,73.86051548500343)
(mosquito,72.38696781805825)
(dxs,72.38696781805825)
(exon,66.94306539426293)
```

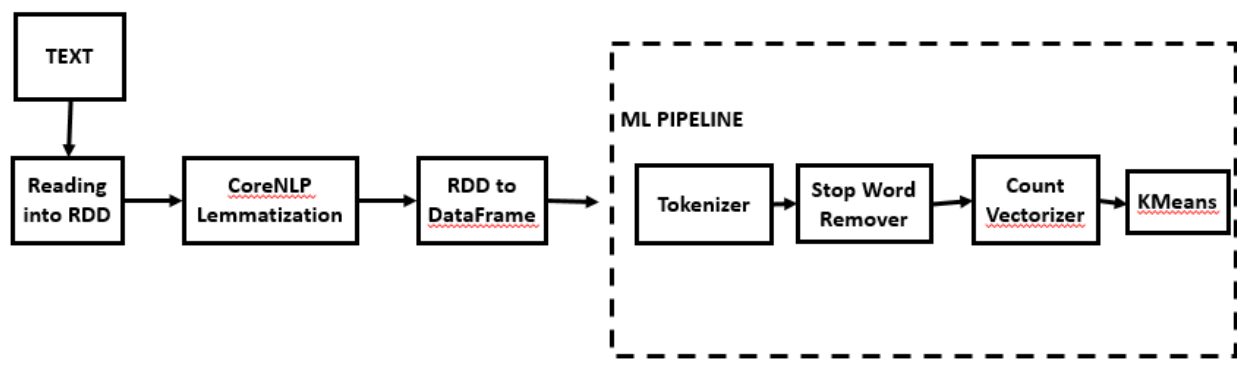
## 4.8 K- Means

K-means is one of the most commonly used clustering algorithms that clusters the data points into a predefined number of clusters.

“k” is the number of desired clusters.

Vector with four elements in which first is location of document, second string is document itself after removing stop words, third is tokens which is array of numbers which related to particular term, and fourth is feature vector and after this topic of that document is printed.

So K-Means categorizes the all documents to some topics.



### Output:

Corpus summary:

Training set size: 7 documents

Vocabulary size: 8066 terms

Preprocessing time: 150.554255247 sec

Finished training KMeans model. Summary:

Training time: 1.341189563 sec

root

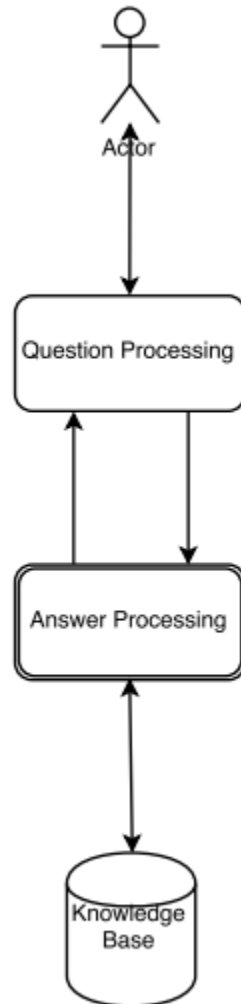
```
|-- location: string (nullable = true)
|-- docs: string (nullable = true)
|-- rawTokens: array (nullable = true)
|   |-- element: string (containsNull = true)
|-- tokens: array (nullable = true)
|   |-- element: string (containsNull = true)
|-- features: vector (nullable = true)
```

()

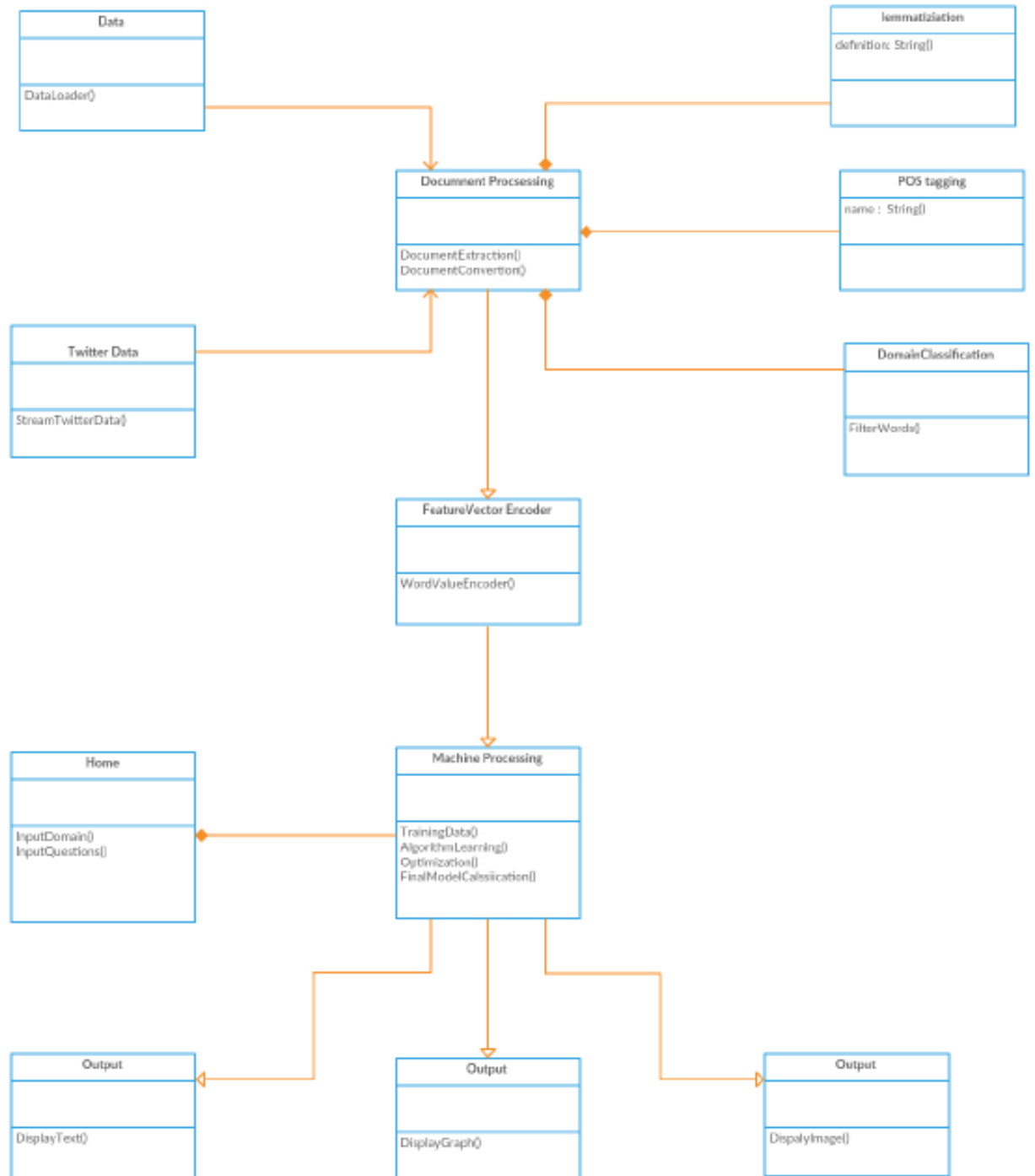
And

## 5. Implementation Specification

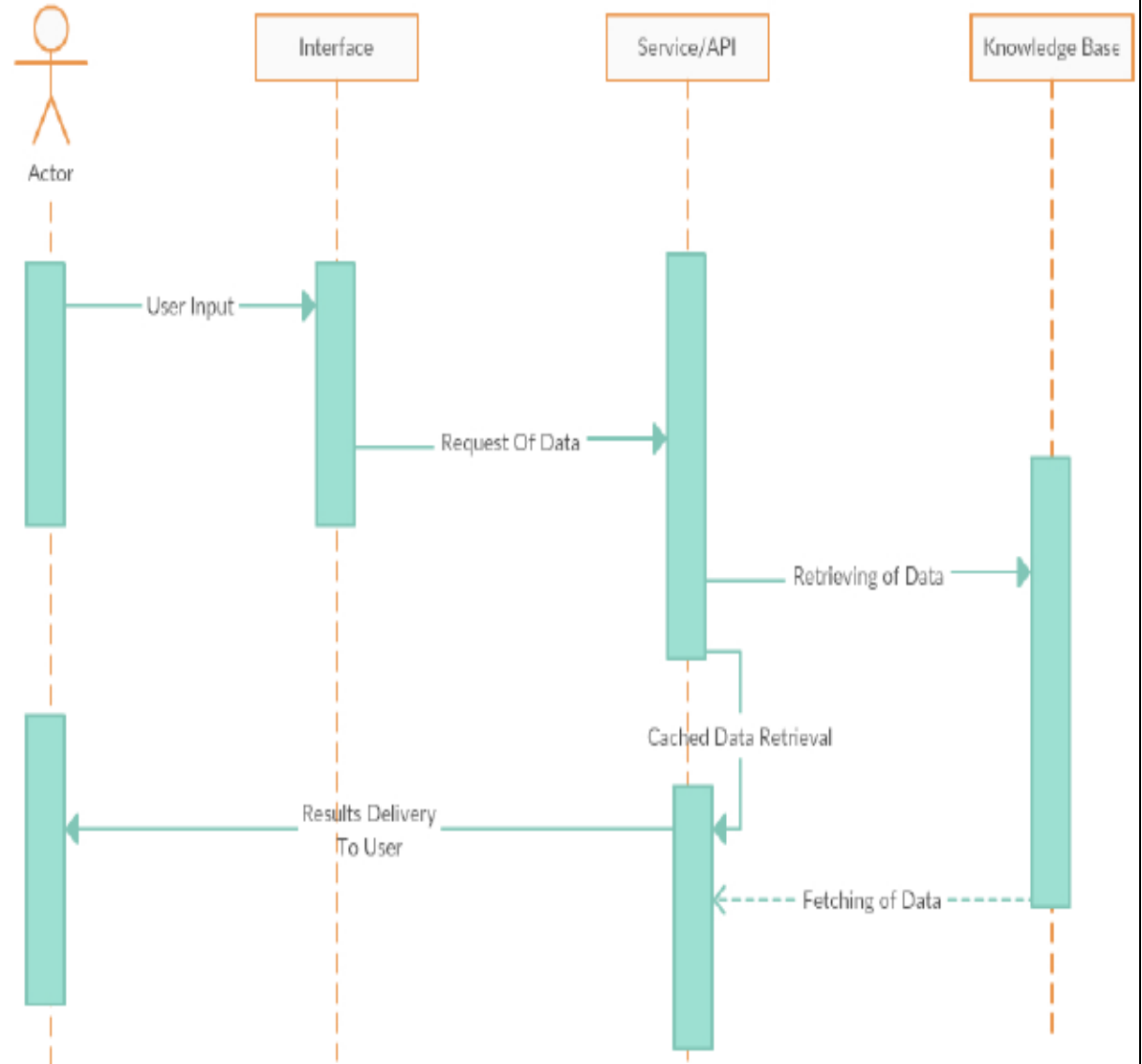
### 5.1 Software Architecture Diagram



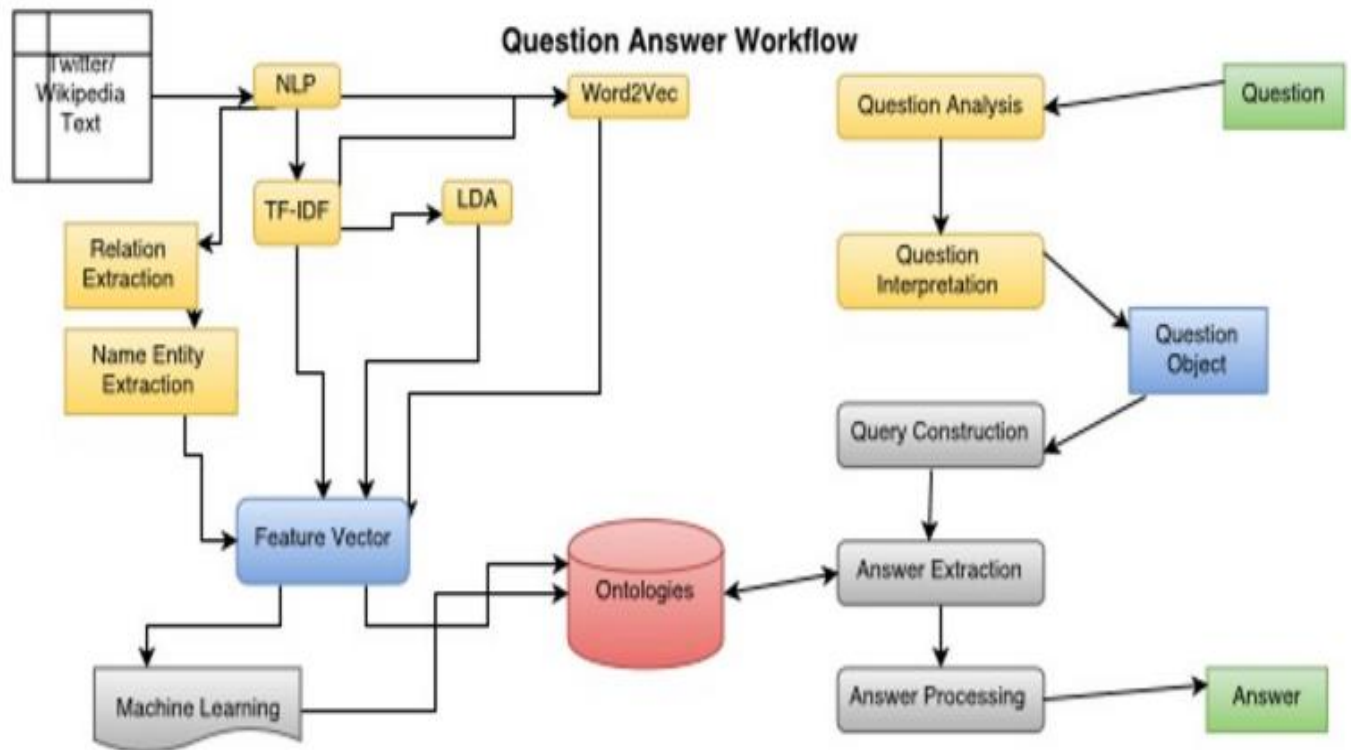
## 5.2 Class Diagram



### 5.3 Sequence Diagram

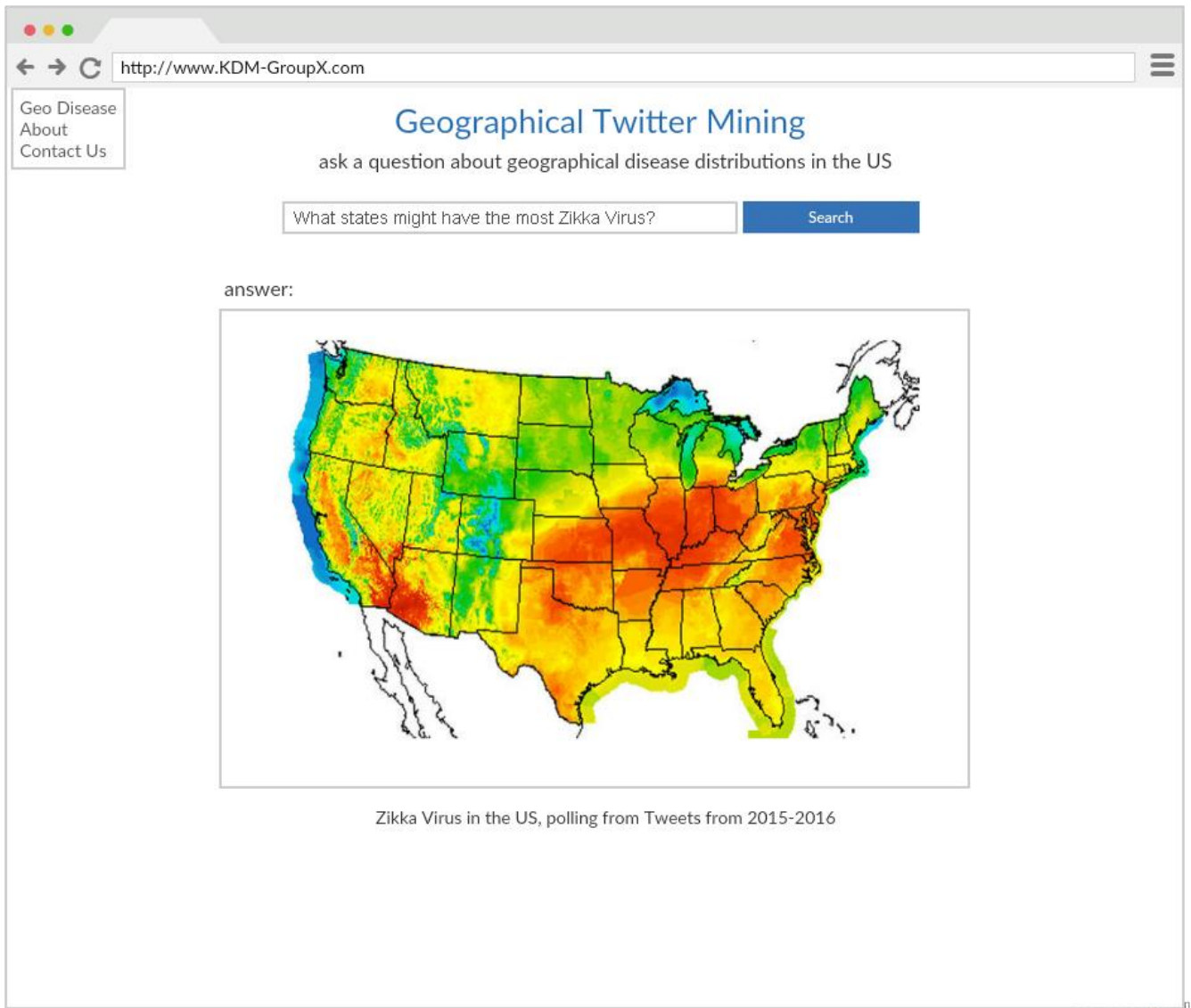


## 5.4 Workflow

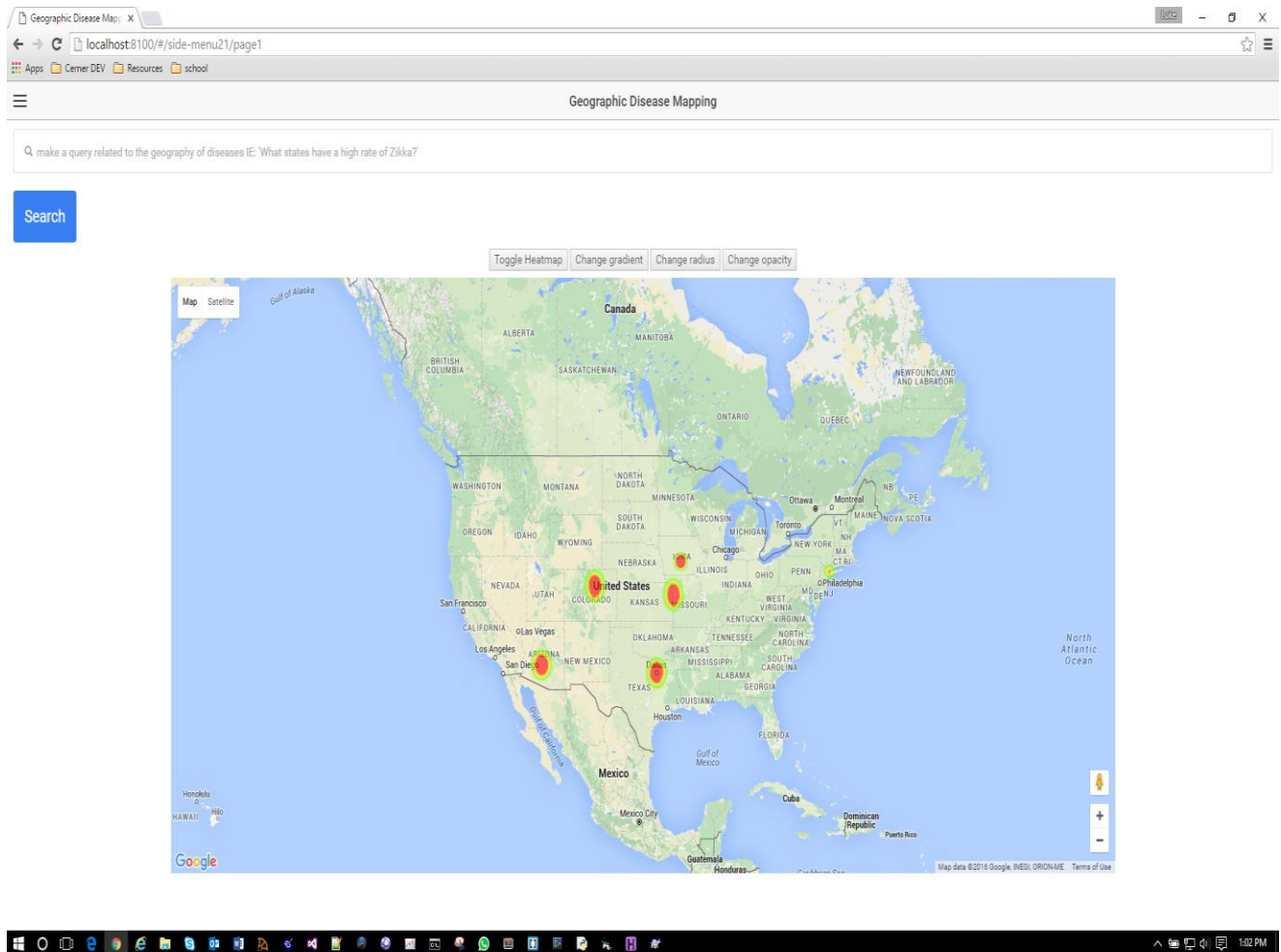




## 5.5 User Interface Wire frame



## 5.6 Expected Output



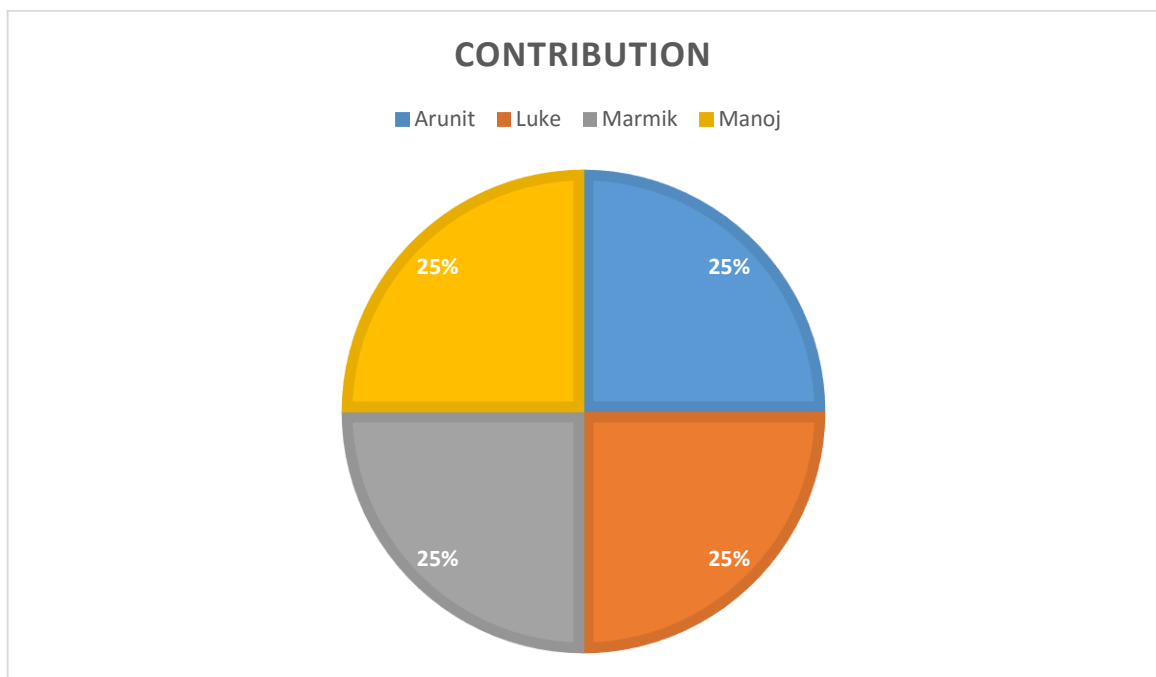
## 5.7 Existing services/features used

AWS, Alchemy API

## 6. Project Management

### 6.1 Contribution of each member

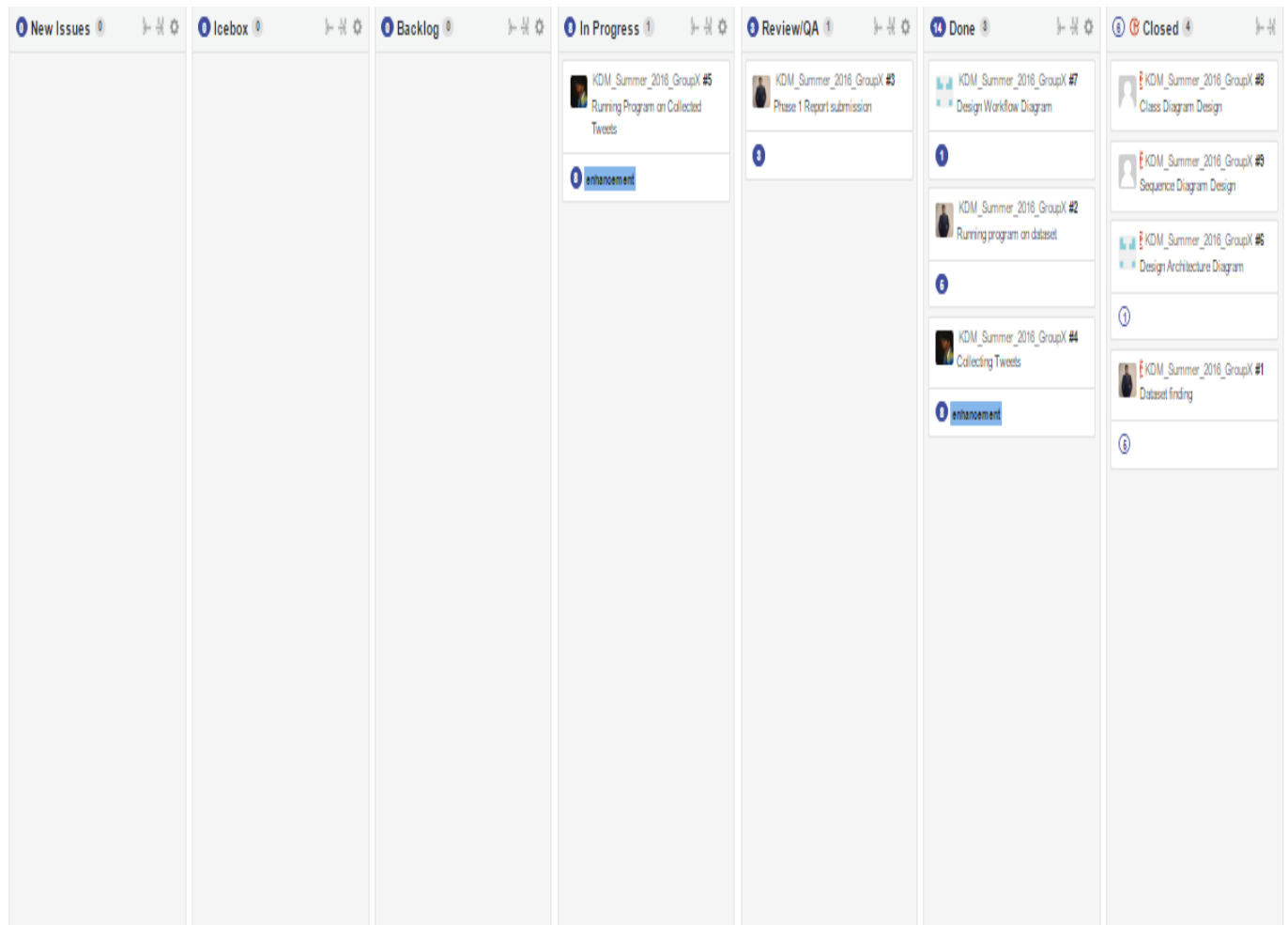
- ❖ **Gupta, Arunit** – Dataset Findings, Running dataset on Program, Report compilation, Word Net, Word Count, NLP, TF-IDF, Documentation.
- ❖ **Patel, Marmikkumar Navinchandra** – collection of tweets, running program on tweets collected, Word2Vec, K-Means.
- ❖ **McDuff, Luke Joseph** – Architecture Diagram design and Workflow design, GUI Design, Wireframes, Front end development.
- ❖ **Ejjirothu, Manoj Prabhakar** – Class Diagram and Sequence Diagram, Topic Discovery using LDA, Feature Vector.

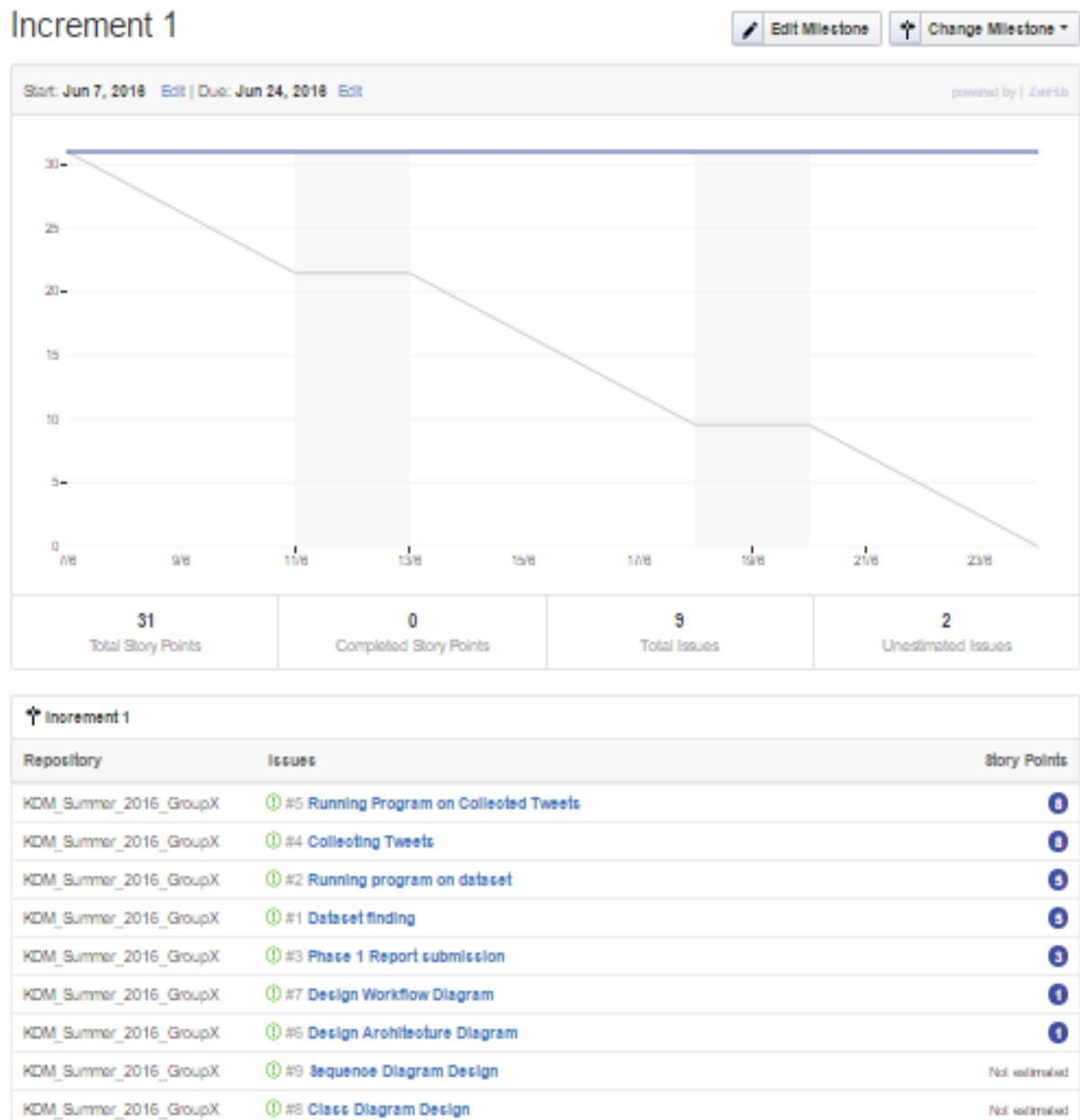


### 6.2 Zen hub and GitHub URL/Statistics

GitHub link: [https://github.com/ljm7b2/KDM Summer 2016 GroupX](https://github.com/ljm7b2/KDM_Summer_2016_GroupX)

## Zen hub





## 6.3 Concerns/Issues

Relating all the technologies and merging them to get final output

## 6.4 Future Work

Implementing the datasets and live data on machine learning and ontologies programs, implementing voice to text search.

## 7. References

- <https://github.com/DiseaseOntology/HumanDiseaseOntology>
- In Class Tutorials
- <https://dev.twitter.com/overview/api>
- <http://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2>
- <http://www.statmt.org/lm-benchmark/1-billion-word-language-modeling-benchmark-r13output.tar.gz>
- <http://ebiquity.umbc.edu/redirect/to/resource/id/351/UMBC-webbase-corpus>
- <http://blog.wolframalpha.com/category/health-med/>
- <http://archive.ics.uci.edu/ml/datasets/Hepatitis>